

Computer Science

Chapter 8: Databases

- 8.1 Database Concepts
- 8.2 Database Management System (DBMS)
- 8.3 Data Definition Language (DDL) and Data Manipulation Language (DML)

By: Mr. Noureddine Tadjerout (MSc)
HOD Computer Science and Virtual Teaching

Learning Objectives

• 8.1 Database Concepts

- Show understanding of the limitations of using a file-based approach for the storage and retrieval of data
- Describe the features of a relational database that address the limitations of a file-based approach
- Show understanding of and use the terminology associated with a relational database model:
Including entity, table, record, field, tuple, attribute, primary key, candidate key, secondary key, foreign key, relationship (one-to-many, one-to-one, many-to-many), referential integrity, indexing
- Use an entity-relationship (E-R) diagram to document a database design
- Show understanding of the normalization process: First Normal Form(1NF), Second Normal Form (2NF) and Third Normal Form (3NF)
- Explain why a given set of database tables are, or are not, in 3NF
- Produce a normalized database design for a description of a database, a given set of data, or a given set of tables

Learning Objectives

- **8.2 Database Management System (DBMS)**

- Show understanding of the features provided by a Database Management System (DBMS) that address the issues of a file based approach Including:
 - ❖ data management, including maintaining a data dictionary
 - ❖ data modelling
 - ❖ logical schema
 - ❖ data integrity
 - ❖ data security, including backup procedures and the use of access rights to individuals / groups of users
- Show understanding of how software tools found within a DBMS are used in practice, Including the use and purpose of:
 - ❖ developer interface
 - ❖ query processor

Learning Objectives

• 8.3 Data Definition Language (DDL) and Data Manipulation Language (DML)

- Show understanding that DBMS carries out all creation / modification of the database structure using its Data Definition Language (DDL)
- Show understanding that the DBMS carries out all queries and maintenance of data using its DML
- Show understanding that the industry standard for both DDL and DML is Structured Query Language (SQL): Understand a given SQL script
- Understand given SQL (DDL) commands and be able to write simple SQL (DDL) commands using a sub-set of commands
 - ❖ Create a database (CREATE DATABASE)
 - ❖ Create a table definition (CREATE TABLE), including the creation of attributes with appropriate data types:
CHARACTER, VARCHAR(n), BOOLEAN, INTEGER, REAL, DATE, TIME
 - ❖ change a table definition (ALTER TABLE)
 - ❖ add a primary key to a table (PRIMARY KEY (field))
 - ❖ add a foreign key to a table (FOREIGN KEY (field) REFERENCES Table (Field))
- Write an SQL script to query or modify data (DML) which are stored in (at most two) database tables :
 - ❖ Queries including SELECT... FROM, WHERE, ORDER BY, GROUP BY, INNER JOIN, SUM, COUNT, AVG
 - ❖ Data maintenance including. INSERT INTO, DELETE FROM, UPDATE

Recap basic Database

- Define a single-table database from given data storage requirements
- Choose and specify suitable data types
- Choose a suitable primary key for a database table
- Perform a query-by-example from given search criteria
- Create a New Database
- Create a New Table
- Adding Data to a Table
- Linking Two Tables

Uses of databases

Databases are very powerful tools used in all areas of computing. It is a key computing skill to be able to organise data, create databases and control data using **query languages**.

One of the main benefits of computer databases is that they make it easy to store information so it is quick and easy to find. For example, if you have music files on your computer, a media **application** like iTunes, Windows Media Player or Google Music organises that data so it is easy for you to quickly search for the artist or songs you want.

Most websites use databases to store data. **Social networking sites** use databases to store data about millions of users, along with photographs and other information about themselves and others

Simply put, a **database is an organized collection of data**.

Databases are very useful in preventing data problems occurring because:

- data is only stored once – no data duplication
- if any changes or additions are made it only has to be done once – the data is consistent
- the same data is used by everyone.

The structure of a database

What is a Databases?

What do you think the word **database** means?



DEFINTION:

A database is a collection of data or information organised in a logical way

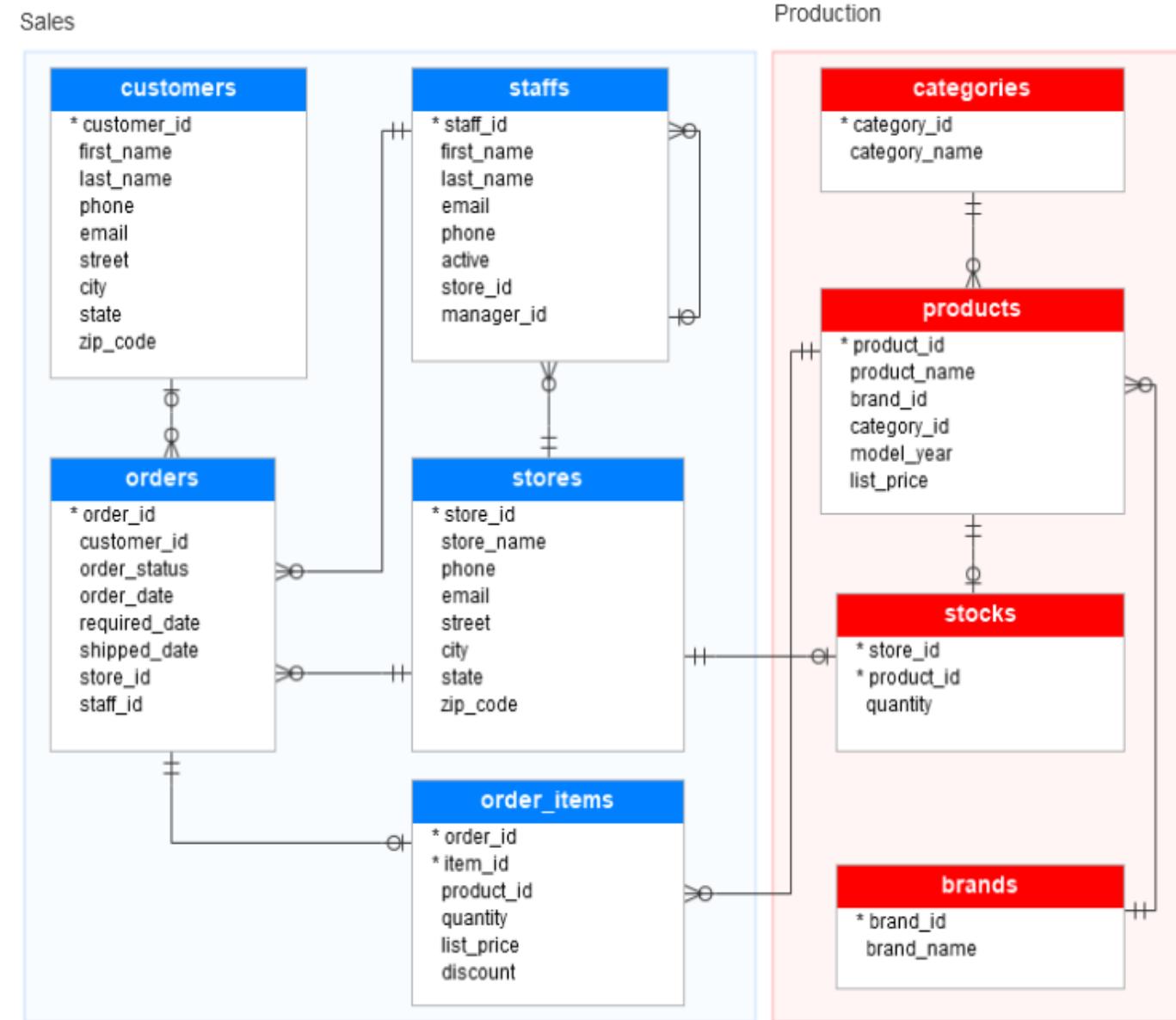
Inside a database, data is stored in **TABLES**, which consists of many **RECORDS** and each record consists of several **FIELDS**.

Databases store information in the form of a **table**. One database can hold **multiple tables** e.g. a school database could have one table for students and another for parents.

What are databases used for?

There are many uses for databases, they are literally everywhere, below are a few examples:

- **Mobile phones** - Contact information stored in a database
- **Videogames** - Stats, scores, ranks and trophies all stored in databases
- **Businesses** - Employee, customer and supplier information all held in databases
- **Hospitals** - Patient health records are held in databases example
- **Law enforcement** - Criminal records are held in databases
- **Education** - Schools have databases for students, parents and staff
- **Social media** - Every member of social media sites have their details stored in a database



Database software

Database software includes **off-the-shelf** software such as **Microsoft Access, Libre Office Base, Oracle, MySQL or NoSQL**.

Databases can also be created and organised using programming languages. Languages like **SQL, Visual Basic** and **Delphi** are used to edit databases. Using **programming languages** means that you can customise a database to do exactly as you want.

Using data

There is so much data being captured online and through smartphones that data is being used in many areas of life. **Big data, open data** and **data mining** are important terms when working with data.

Big data

Big data describes extremely large sets of data. It includes data gathered from many different sources that is then analysed. Big data is often used for making predictions based on patterns that can be seen in the data.

Data mining

Data mining is a term used to describe analysing large amounts of data to predict future events and trends. As there is so much data now available, people who are able to analyse and understand data are going to be well placed to shape the development of technology.

Open data

Many organisations now share large sets of data freely. Organisations like the government, local councils, and world organisations (like the United Nations) make data freely available. This means that anyone can look at and analyse the data. Journalists often use open data sets to form the basis of news articles. Open data projects are used to collaborate and share data around the world. For example, the Skynet project allows amateur and professional astronomers around the world to share information about the stars, planets, satellites and meteors.

Data security

We put a lot of trust in companies when we give them our personal details. They have a legal obligation to ensure that our data is kept in a secure centralised **database**. Data **encryption** is a necessity for any database containing personal data.



Photo courtesy by BBC Bitesize

Creating a database

When creating a database think about what data you need to store. A database is essentially a collection of details about different items. You could create a database about pretty much anything.

Entities

When you build a database you are organising data about **entities**. An entity is any item that has its **attributes** stored as data. An entity could be anything, example: a book, a person, a film, a country or a football team.

Attributes

The details about entities are called attributes. A person is an entity with attributes including age, height and nationality, among many others. When you design a database you need to think about which attributes you want to store. For example, the attributes of a film could include title, duration, certificate, rating, gender, cast, director and year of creation

In a database of hotels, an individual hotel is the entity, and the attributes could be ranking, awards, location, photos, ratings and ID number.



Photo courtesy by BBC Bitesize

Database structure:

A **database** is organised using a set of key components. These include:

- **entities** - each recorded item
- **attributes** - details about the entity
- **field** - columns used to capture attributes
- **record** - one row of details about an entity
- **table** - a set of fields and records
- **primary key** - unique number for an entity

This is an example table of a **flat-file database**. The **entities** are films and the **attributes** are details about the films:

The diagram illustrates a flat-file database table with the following structure:

Fields (column headings)						
	FilmID	Title	Duration	Certificate	Rating	Genre
Unit of data	1	Zombie Attack	1:32:00	18	***	Horror
Primary key	2	True Love	1:28:00	12	****	Romance
Records (rows)	3	Mission: Pluto	2:19:00	15	**	Sci-Fi

Annotations point to specific parts of the table:

- Unit of data:** Points to the first row (FilmID 1).
- Primary key:** Points to the primary key column (FilmID).
- Records (rows):** Points to the first record (FilmID 1, "Zombie Attack").

A pink callout at the bottom right points to the word "Table".

Database structure:

Table

The table contains all of the **fields** and the **records** for one type of entity. A database may contain more than one table.

Records

Records contain a collection of data for each entity, usually recorded as a row in the table.

Fields

The column headings are called the fields. Each field contains a different attribute. For every entity, a unit of **data** will be entered into each field. Each column might require different data types. For example, the 'Title' column will require data entered as text and the 'Certificate' column will need data entered as numbers.

Unit of data

Each individual piece of data entered is a unit of data. These units are also called data **elements**.

The primary key:

Contains a unique identifier for each record. To make each record in a database unique we normally assign them a primary key. Even if a record is deleted from a database, the primary key will not be used again. The primary key can be automatically generated and will normally just be a unique number or mix of numbers and letters.

Data types

The actual units of data that are entered into a **database** give the **attributes** for each **entity**. These units of data are also called **data elements**.

When you create a database you need to set data types for each **field**. For example, in a film database you might need alphabetical characters for 'Titles', but numbers for 'Duration'. Fields are usually restricted to a certain data type.

Data typing is a way of classifying data values that have common properties. Different kinds of data values also need different amounts of **memory** to store them, and have different operations that can be performed upon them.

Common data types:

The most commonly-supported data types are:

integers (whole numbers), for example: 4, 27, 65535

floating point numbers (with decimal points, sometimes called real numbers, or floats), for example: 4.2, 27.4, 56.8

characters, for example: a, F, 3, \$, £, #

character strings (ordered sequences of characters), for example: abc, def456, 3erf78!@

Boolean values, for example: 'True' or 'False'

Data types

SQL data typing:

In the database programming language **SQL**, the data types can be set as follows:

Data type	Explanation
char(size)	Fixed-length character string. Size is specified in brackets.
varchar(size)	Variable-length character string. Maximum size is specified in brackets.
integer(size)	Number value with a maximum number of column digits specified in brackets.
date	Date value
float(size,d)	Number value with a maximum number of digits of 'size' total, with a maximum number of 'd' digits to the right of the decimal.
Boolean	True/False values.

Data structure

The structure of the **database** is also called the **schema** or **dictionary**. When you design a database you need to create a schema to explain what type of data is being stored. There are key areas to consider about the data:

- data type** - the data types are used in each **field**
- field size** - the maximum or minimum size of entry
- validation** - the rules for accepting data
- key field** - the field which is the **primary key**

Data structure

Before creating a database, you need to think about its purpose. What is the database going to be used for? What searches might be performed on it?

The diagram illustrates the structure of a database table. At the top, a blue box labeled "Fields (column headings)" has arrows pointing to the column headers: FilmID, Title, Duration, Certificate, Rating, and Genre. Below the headers is a row of data with three numbered callouts: "Unit of data" points to the first row; "Primary key" points to the "FilmID" column; and "Records (rows)" points to the three rows of data. The data rows contain the following information:

FilmID	Title	Duration	Certificate	Rating	Genre
1	Zombie Attack	1:32:00	18	***	Horror
2	True Love	1:28:00	12	****	Romance
3	Mission: Pluto	2:19:00	15	**	Sci-Fi

Table

In this film table example, the structure of the table would be:

Field name	Data type	Size in bytes	Primary key?
Film ID	Integer	2	Yes
Title	Text	20	No
Certificate	Integer	2	No
Genre	Text	20	No

Photo courtesy by BBC Bitesize

Database integrity

The integrity of the **database** relates to the data being valid, accurate and consistent.

Make sure your database has been designed as you want before data is entered - it is usually difficult to change a database structure after data has been entered.

A phrase database designers commonly think about is "garbage in, garbage out". This means that if you do not design your database properly, and do not take in useful information, you will not get useful information out of it.

There are two main checks to make sure a database is accurate:

verification
validation



Validation checks that the correct type of data is entered, whereas verification checks that the data is actually the data you want.

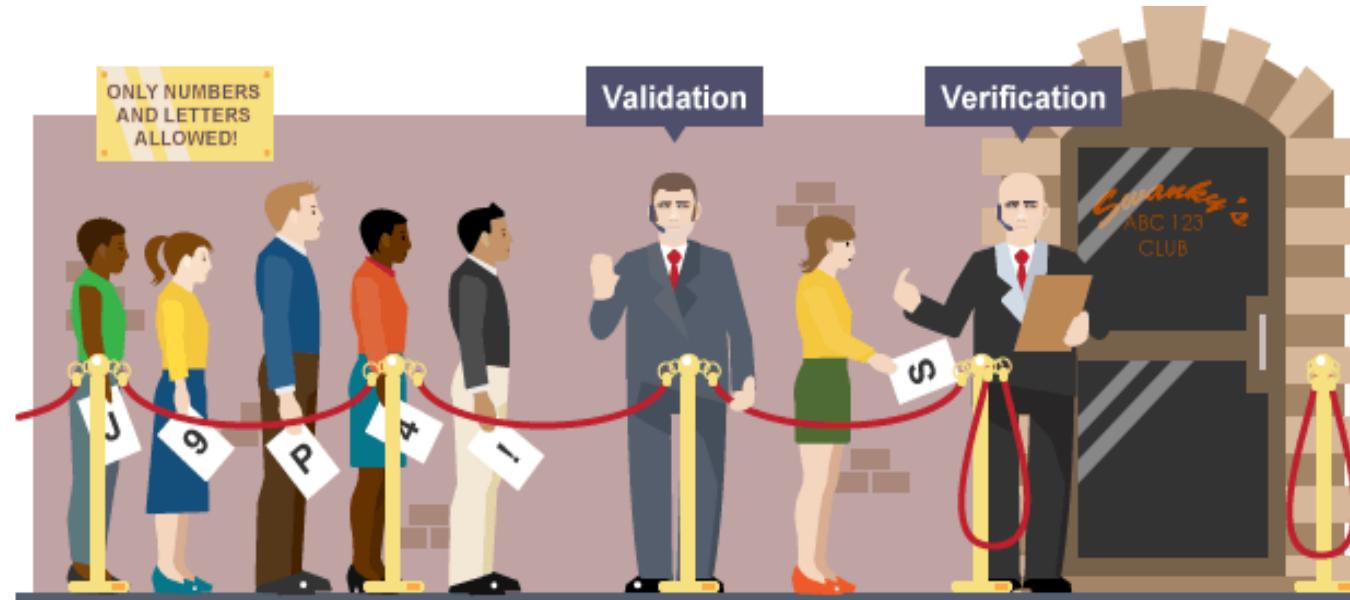


Photo courtesy by BBC Bitesize

Database integrity

For example, validation checks that a postcode has been entered in the correct format and verification checks that the postcode being entered is the correct postcode.

A good way of doing a verification check is by having two people entering the same data. A verification check will see if both sets are the same. If there are differences, the check will bring up an error message and ask for the data to be re-entered

Validation



Database fields should have **validation rules** to make sure the data entered follows the expected format.

Photo courtesy by BBC Bitesize

Database integrity

Validation

There are different types of validation checks a database can run:

Validation type	How it works	Example
Check digit	The last couple of digits can be used as a 'check sum' that can detect if errors have occurred	Bar code readers in shops
Type check	Checks the data is in the right format	Numbers in currency cell must be a monetary value with two decimal points
Length check	Checks the data is an acceptable length	A PIN for online banking needs to be four (or six) characters long
Lookup table	Checks that the value provided matches an item in a set list	A limited set of values, such as the seven days of the week
Presence check	Checks that data has been entered into a field	In most databases a key field cannot be left blank
Range check	Checks that a value falls within the specified range	An online gift certificate purchase must be more than or equal to £5 but less than or equal to £50
Spell check	Looks up words in a dictionary	A search engine often recommends a correct spelling if a word is spelt wrong
Input mask	Checks that data has been entered with the correct amount of characters and/or numbers	National insurance numbers need to be in the format: YY XX XX XX Y, where Y = letter and X = number
Duplicate	Checks that a value has not been repeated	A primary key value can only be entered once

The computer will use these rules to check whether the data entered is correct. If it is invalid an error message will be created

What are databases used for?

Watch video for: Database Tutorial for Beginners?:

<https://www.youtube.com/watch?v=wR0jg0eQsZA>



2.3.3-The structure of a database

Finding the Information you need

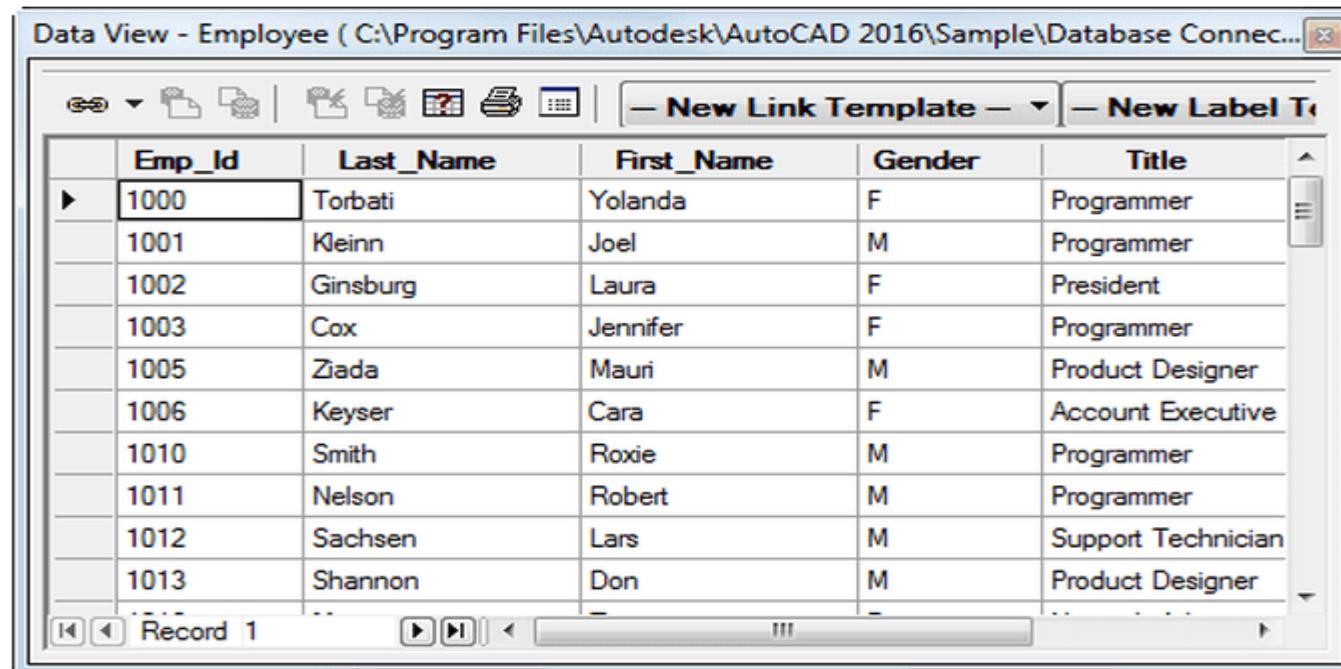
In a database records can be:

- **Searched** – finding a particular piece of information.
- **Sorted** – records in a database are sorted in a particular order.

The structure of a database

Databases store information in the form of a **table**. One database can hold **multiple tables** e.g. a school database could have one table for students and another for parents.

Here is an example database table, this table is storing employee data



A screenshot of the AutoCAD Data View window titled "Data View - Employee". The window displays a table of employee data with the following columns: Emp_Id, Last_Name, First_Name, Gender, and Title. The table contains 13 records. The first record (Emp_Id 1000) is currently selected. The last record (Emp_Id 1013) is visible at the bottom. The window has standard toolbar icons and navigation buttons at the bottom.

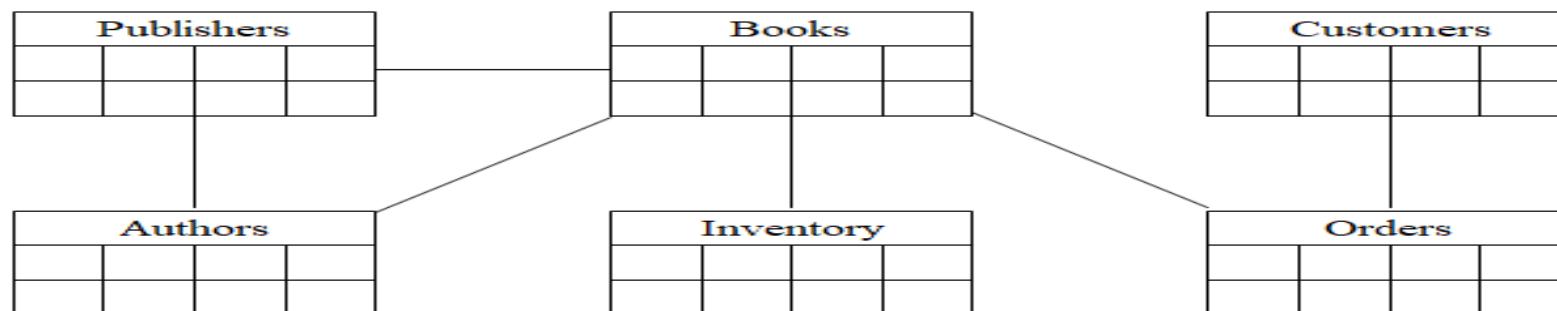
	Emp_Id	Last_Name	First_Name	Gender	Title
▶	1000	Torbati	Yolanda	F	Programmer
	1001	Kleinn	Joel	M	Programmer
	1002	Ginsburg	Laura	F	President
	1003	Cox	Jennifer	F	Programmer
	1005	Ziada	Mauri	M	Product Designer
	1006	Keyser	Cara	F	Account Executive
	1010	Smith	Roxie	M	Programmer
	1011	Nelson	Robert	M	Programmer
	1012	Sachsen	Lars	M	Support Technician
	1013	Shannon	Don	M	Product Designer

The structure of a database

Table

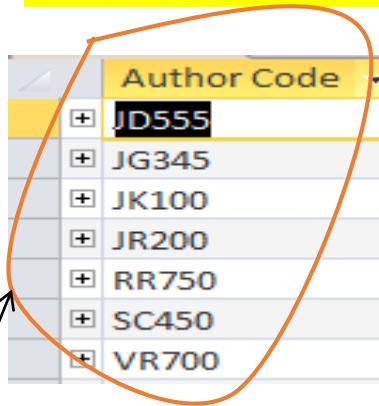
	Author Code	Author Name	Nationality	DOB	University
[+]	JD555	James Dashner	American	26/11/1972	Brigham Young University
[+]	JG345	John Green	American	24/08/1977	Kenyon College
[+]	JK100	J. K. Rowling	British	31/07/1965	Exeter University
[+]	JR200	J. R. R. Tolkien	British	03/01/1892	University of Oxford
[+]	RR750	Rick Riordan	American	05/06/1964	University of Texas
[+]	SC450	Suzanne Collins	American	10/08/1962	New York University
[+]	VR700	Veronica Roth	American	19/08/1988	Northwestern University

A Database with Multiple Tables



Field (Column)

below you have a total of 5 fields

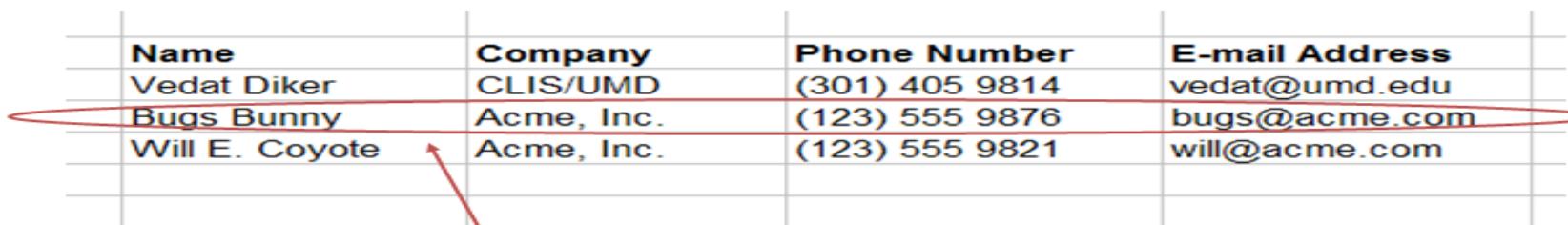


Author Code	Author Name	Nationality	DOB	University
JD555	James Dashner	American	26/11/1972	Brigham Young University
JG345	John Green	American	24/08/1977	Kenyon College
JK100	J. K. Rowling	British	31/07/1965	Exeter University
JR200	J. R. R. Tolkien	British	03/01/1892	University of Oxford
RR750	Rick Riordan	American	05/06/1964	University of Texas
SC450	Suzanne Collins	American	10/08/1962	New York University
VR700	Veronica Roth	American	19/08/1988	Northwestern University

a field

Record (Row)

below you have total of 3 records



Name	Company	Phone Number	E-mail Address
Vedat Diker	CLIS/UMD	(301) 405 9814	vedat@umd.edu
Bugs Bunny	Acme, Inc.	(123) 555 9876	bugs@acme.com
Will E. Coyote	Acme, Inc.	(123) 555 9821	will@acme.com

a record

Primary Key

Customers

Customer ID	Name	Company	Phone Number	E-mail Address
6273	Vedat Diker	CLIS/UMD	(301) 405 9814	vedat@umd.edu
3245	Bugs Bunny	Acme, Inc.	(123) 555 9876	bugs@acme.com
1324	Will E. Coyote	Acme, Inc.	(123) 555 9821	will@acme.com

A red circle highlights the "Customer ID" column header. A red arrow points from the text "primary key field" to the bottom of the "Customer ID" column header.

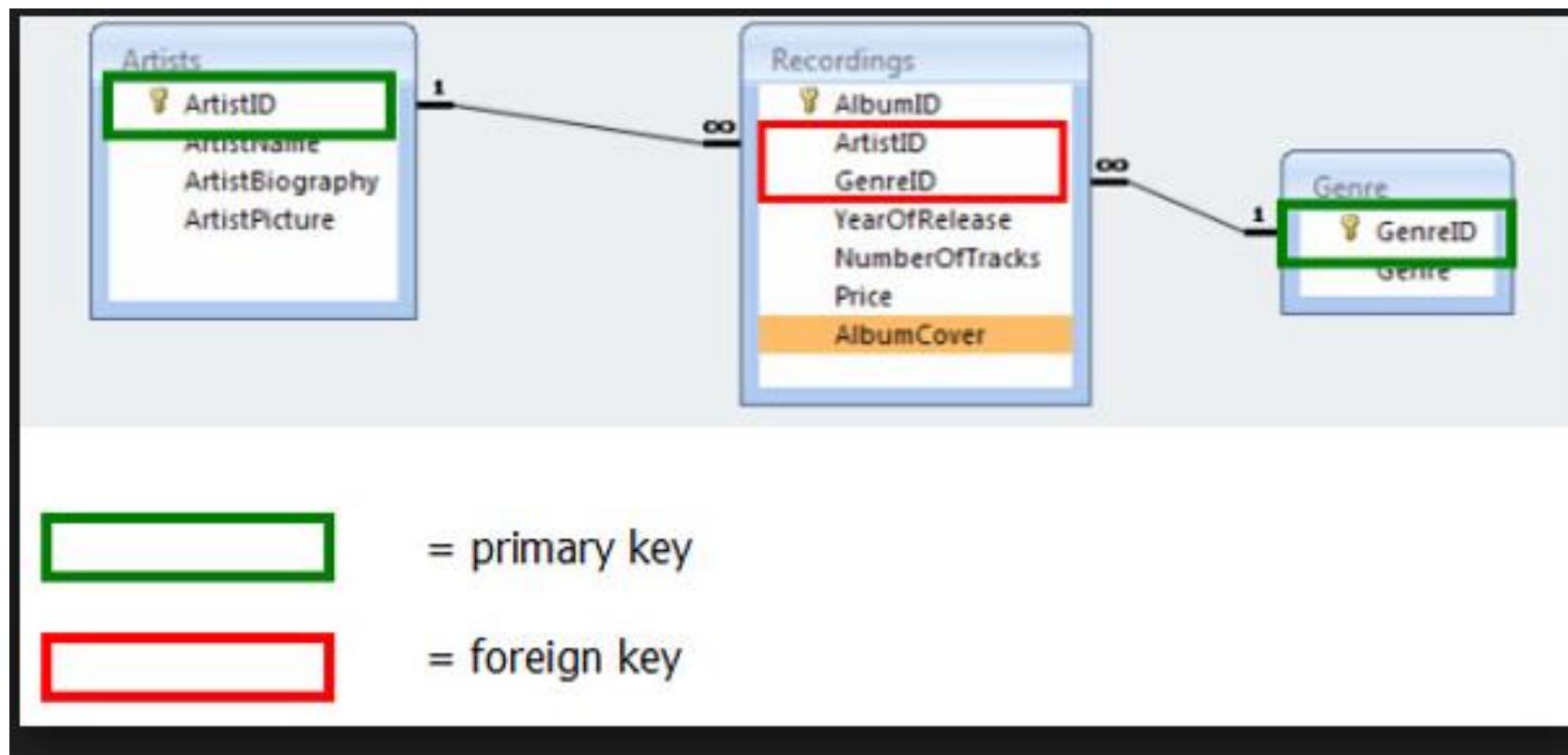
primary key field

Primary key is a unique identifier of records in a table.

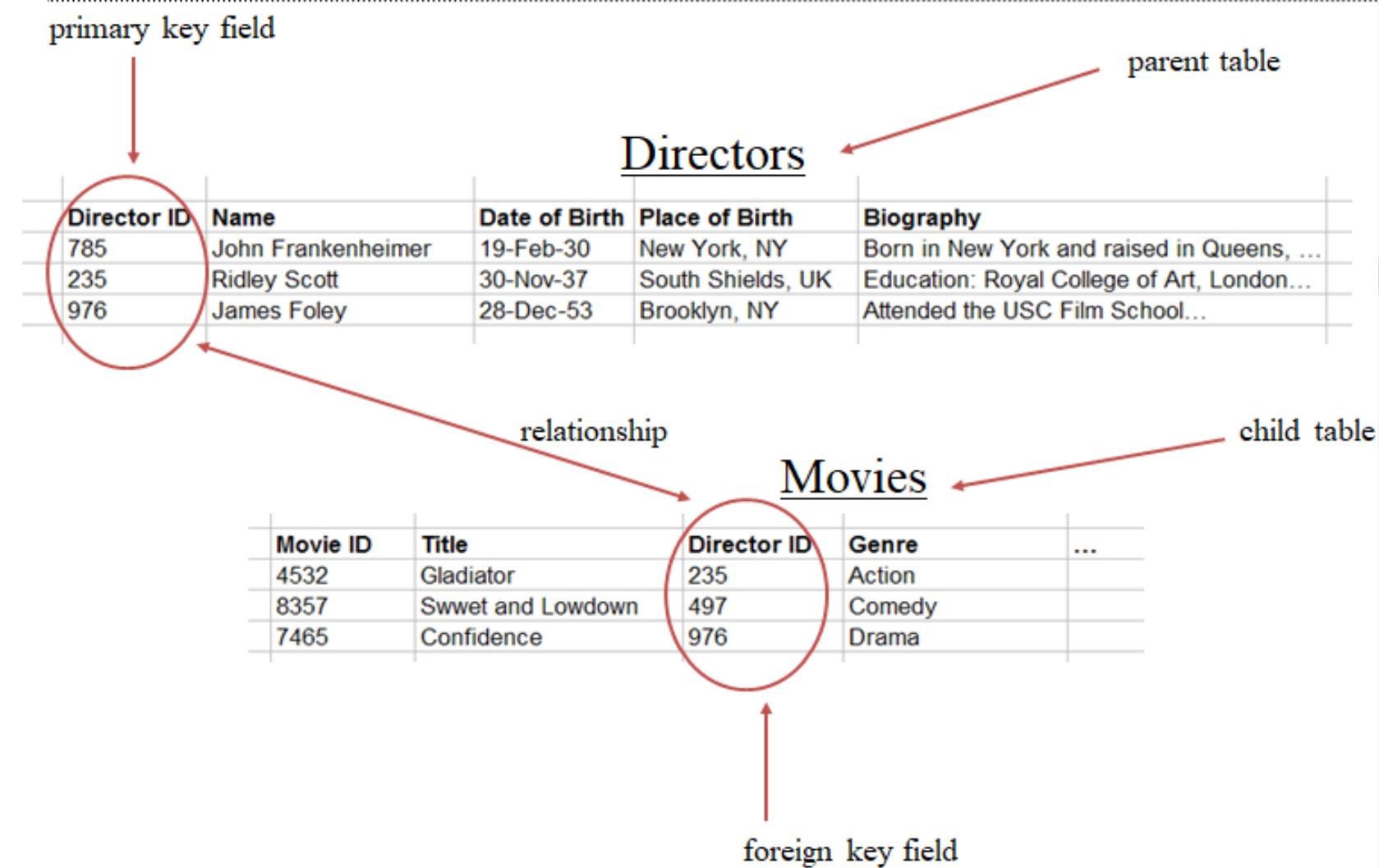
Primary key values may be generated manually or automatically.

Foreign Key

A **foreign key** is a column or group of columns in a relational **database** table that provides a link between data in two tables. It acts as a cross-reference between tables because it references the primary **key** of another table, thereby establishing a link between them



Foreign Key



The structure of a database

Data Types:

When setting up a database, each field must be given a name and a "Data type". If you are using Microsoft Access to set up your database, the process will look like this:

Available data types:

- Alphanumeric (Text, Memo)
- Numeric (Number, Currency, etc.)
- Date/Time
- Boolean (Yes/No)

Computer program	Access
Integer	Number/Integer
Real	Number
Char	Text/field length 1
String	Text
Boolean	yes/no

Table1		
	Field Name	Data Type
!	Emp_Id	AutoNumber
	Last Name	Text
	First Name	Text
	Gender	Text
	Title	Text

The structure of a database

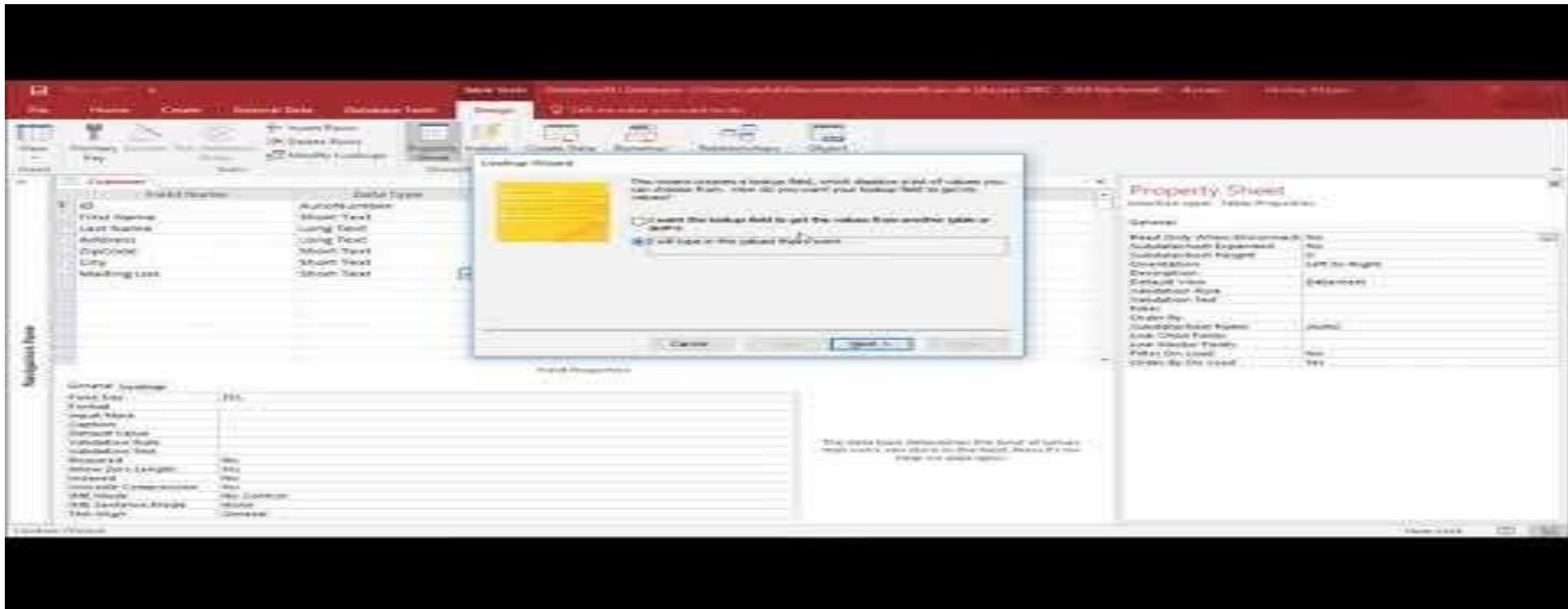
Available data types with description:

Data Type	Description
Text	Stores a short amount of text e.g a name, address, occupation
Integer	Stores a whole number e.g Year group = 10
Dates/ Times	Stores a date or time in a specified format e.g. 09/02/1988
Currency	Stores a monetary value with the currency symbol e.g. \$44
Boolean (Yes / no)	Able to store one of two values e.g. yes or no, Male or Female
Real	Stores a number with a decimal point e.g. 5.6

Practical use of a database

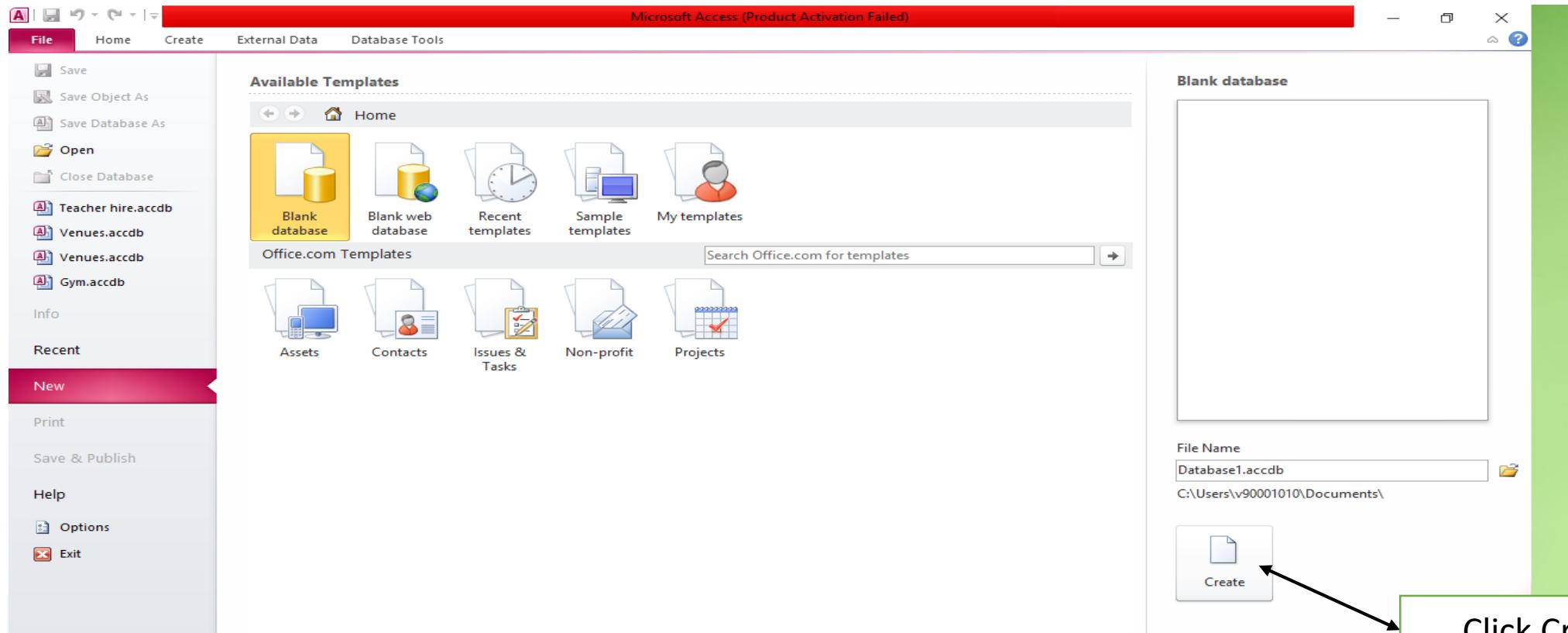
Watch video for Creating a Database:

<https://www.youtube.com/watch?v=pbtoCktIX1Y>

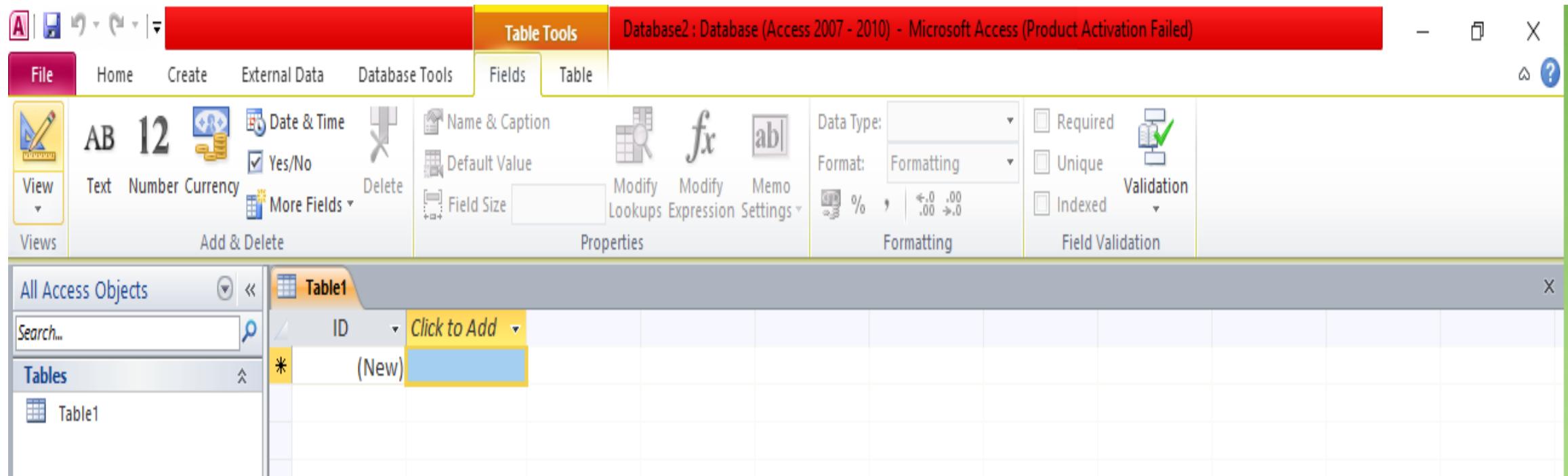


Microsoft Access and complete the tasks described below in next slide how to create a database.

Practical Activity : Creating a Database



You'll then see this Access screen:



1. Click on the **View** button (see Figure 11).

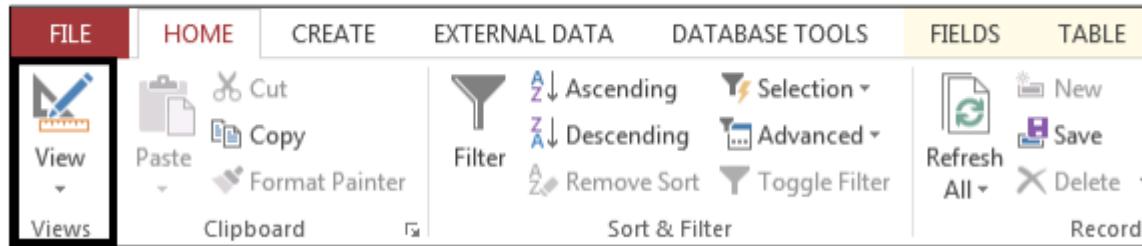


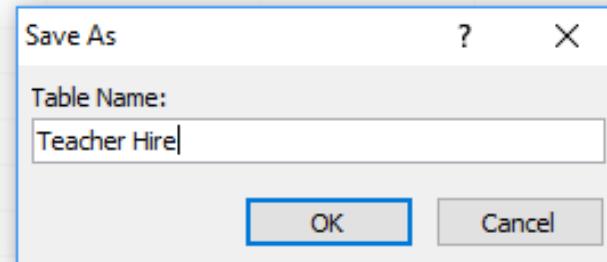
Figure 11 - View Button

2. In the menu that appears, select **Design View** (see Figure 12).



Figure 12 - Design View

3. Enter the name of the table that you are about to create in the Save As dialogue box (see Figure 13).



4. Click **Ok**

A | | Table Tools | Database3 : Database (Access 2007 - 2010) - Microsoft Access (Product Activation Failed)

File Home Create External Data Database Tools Fields Table

View

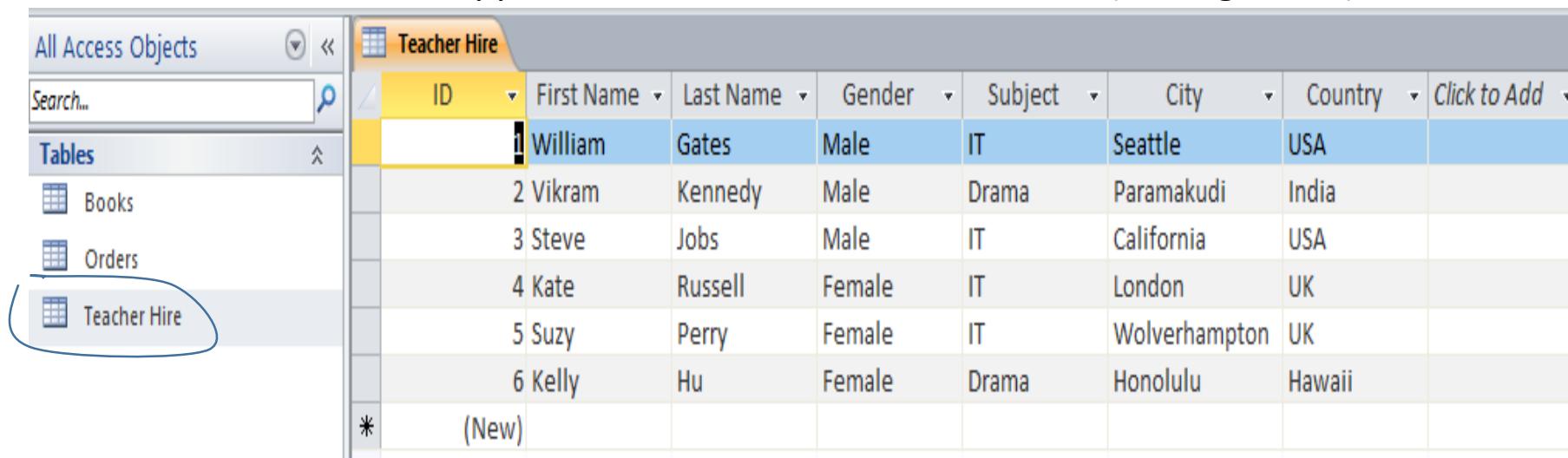
All Access Objects TeacherHire

	ID	First Name	Last Name	Gender	Age	School Name	Subject	City	Country	Click to Add
	1	Noureddine	Tadjerout	M	35	BST	Computer Science	London	UK	
*	(New)									

5. Enter the Field Names and Data Types (see Figure 14).

Field Name	Data Type
ID	AutoNumber
First Name	Text
Last Name	Text
Gender	Text
Subject	Text
City	Text
Country	Text

To open the table again, **double-click Teacher Hire table** that you want to open. The name of the table will appear on the left area of the window (see Figure 15).



The screenshot shows the Microsoft Access interface. On the left, the 'All Access Objects' navigation pane is visible, featuring a search bar and sections for 'Tables', 'Books', 'Orders', and 'Teacher Hire'. The 'Teacher Hire' item is highlighted with a blue oval. The main workspace displays the 'Teacher Hire' table in Datasheet view. The table has columns: ID, First Name, Last Name, Gender, Subject, City, Country, and a 'Click to Add' button. Six rows of data are shown, starting with ID 1 (William Gates, Male, IT, Seattle, USA) and ending with ID 6 (Kelly Hu, Female, Drama, Honolulu, Hawaii). A new row is indicated at the bottom with an asterisk (*) and '(New)'.

ID	First Name	Last Name	Gender	Subject	City	Country	Click to Add
1	William	Gates	Male	IT	Seattle	USA	
2	Vikram	Kennedy	Male	Drama	Paramakudi	India	
3	Steve	Jobs	Male	IT	California	USA	
4	Kate	Russell	Female	IT	London	UK	
5	Suzy	Perry	Female	IT	Wolverhampton	UK	
6	Kelly	Hu	Female	Drama	Honolulu	Hawaii	
*	(New)						

4 - Entering Data into the Table 1

Press Tab to move from field to field or click in a cell.

The screenshot shows the Microsoft Access interface with the 'Table Tools' ribbon selected. On the left, the 'All Access Objects' navigation pane is open, showing 'Tables' and 'Teacher Hire'. The main area displays the 'Teacher Hire' table with the following data:

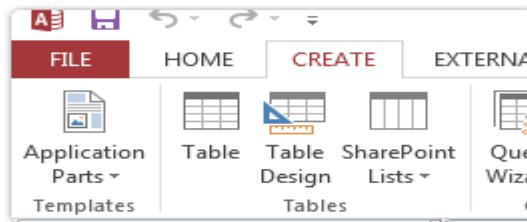
ID	First Name	Last Name	Gender	Subject	City	Country	Click to Add
1	William	Gates	Male	IT	Seattle	USA	
2	Vikram	Kennedy	Male	Drama	Paramakudi	India	
3	Steve	Jobs	Male	IT	California	USA	
4	Kate	Russell	Female	IT	London	UK	
5	Suzy	Perry	Female	IT	Wolverhampton	UK	
6	Kelly	Hu	Female	Drama	Honolulu	Hawaii	

The cell for the new entry ('New') is highlighted with a yellow border.

First Name	Last Name	Gender	Subject	City	Country
William	Gates	Male	IT	Seattle	USA
Vikram	Kennedy	Male	Drama	Paramakudi	India
Steve	Jobs	Male	IT	California	USA
Kate	Russell	Female	IT	London	UK
Suzy	Perry	Female	IT	Wolverhampton	UK
Kelly	Hu	Female	Drama	Honolulu	Hawaii
Azuma	Nelson	Male	P.E	Accra	Ghana
Kevin	Baxter	Male	IT	Birmingham	UK
Usain	Bolt	Male	P.E	Trelawny	Jamaica
Ainsley	Harriott	Male	Cookery	London	UK
Dayana	Mendoza	Female	Drama	Caracas	Venezuela
Ann	Widdecombe	Female	Literature	Somerset	UK
Michelle	Obama	Female	Law	Washington	USA
Natália	Guimarães	Female	Drama	Juiz de Fora	Brazil

Entering Data to create another Table

To create a new table, go to the Create tab. You'll see it to the right of the File tab. Click Table to add another table.



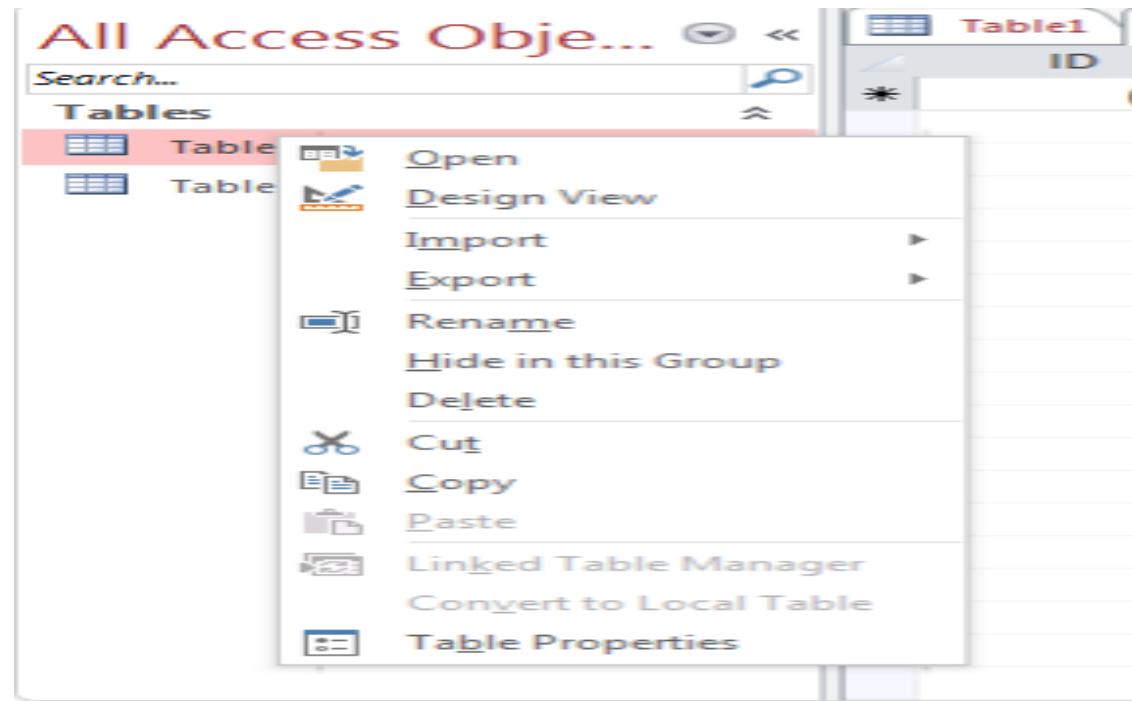
Access creates one table for you by default, so when we do this, we'll have two tables, as shown below.

A screenshot of the Microsoft Access application. The ribbon is visible at the top, with the FILE tab selected. Below the ribbon is the Quick Access Toolbar. The main area shows the Database window. On the left, the 'All Access Objects' pane displays 'Tables' with 'Table1' and 'Teacher Hire' listed. In the center, there are two tables: 'Teacher Hire' and 'Table1'. The 'Table1' tab is active, showing a single row with columns 'ID' and '(New)'. A yellow callout box points to the 'Click to Add' button in the first column of the table grid. The 'Table Tools' tab is selected in the ribbon.

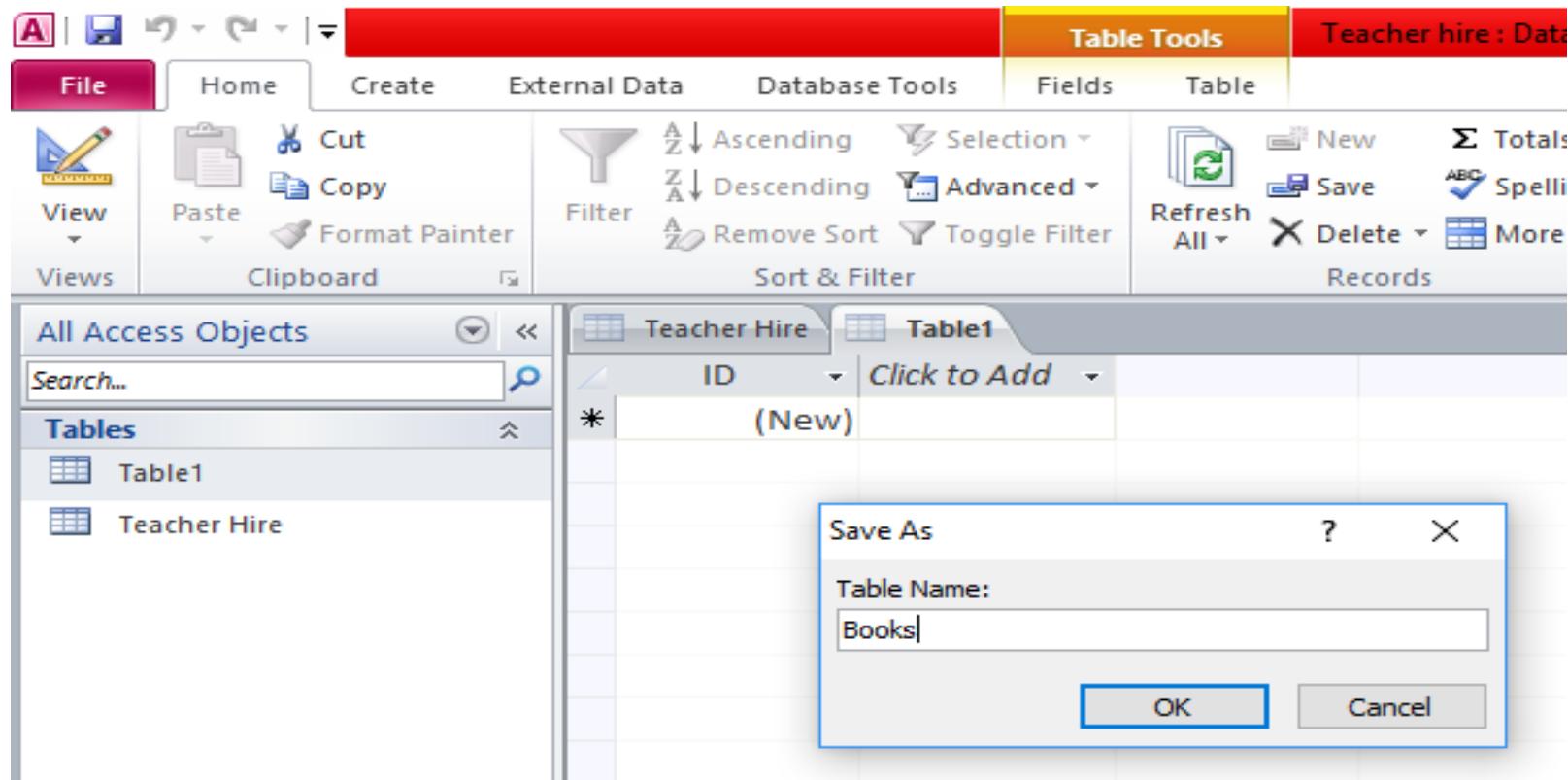
You can click on the Table1 or Teacher Hire tab to navigate between tables.

On the left hand side of the screen, right click Table1 and select **Design View.**

This is where you can design the table by determining what fields you want to store in the table.



Access will ask you to name the table. Enter a table name (**Books**), then click **OK**



I named ours Books

The screenshot shows the Microsoft Access 2010 interface with the 'Table Tools' ribbon selected. The 'Design' tab is active. In the center, the 'Books' table is displayed in a grid format with columns for 'Field Name', 'Data Type', and 'Description'. The first row shows a field named 'ID' with a data type of 'AutoNumber'. On the left, the 'Tables' pane shows 'Books' and 'Teacher Hire' as available objects. At the bottom, the 'Field Properties' pane is open for the 'ID' field, showing settings like 'Field Size: Long Integer', 'Format: Increment', and 'Indexed: Yes (No Duplicates)'. A note in the properties pane states: 'A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.'

Field Name	Data Type	Description
ID	AutoNumber	

Field Properties

General	Lookup
Field Size	Long Integer
New Values	Increment
Format	
Caption	
Indexed	Yes (No Duplicates)
Smart Tags	
Text Align	General

A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.

In **Design View**, you can go ahead and name the table columns and apply constraints over them in the Field Properties at the bottom of the above pictured window. By clicking on **ID**, you can also add your Primary key.

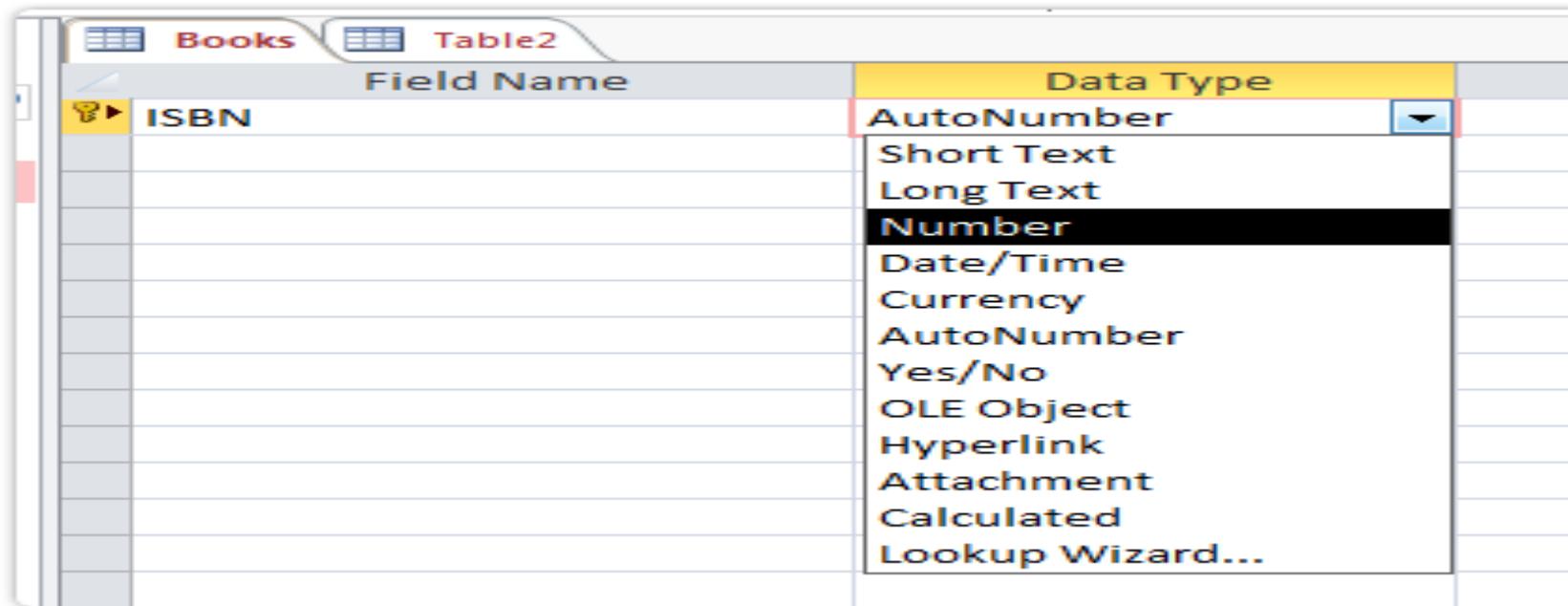
We're going to make our Primary Key **ISBN**.

We'll write that in the Field Name. This is the name of the field/column.

The screenshot shows the Microsoft Access ribbon with the 'Table Tools' tab selected. In the 'Primary Key' section of the ribbon, the 'Primary Key' icon is highlighted. On the left, the navigation pane shows 'All Access Objects' and 'Tables' with 'Books' selected. The main area displays a table with one row. The first column is labeled 'Field Name' and contains the value 'ISBN'. The second column is labeled 'Data Type' and contains the value 'AutoNumber'.

Field Name	Data Type
ISBN	AutoNumber

Next, we'll "tab over" to **Data Type** and specify the type of data to be entered into the **ISBN** field, by pressing the Tab key. Select "Number".



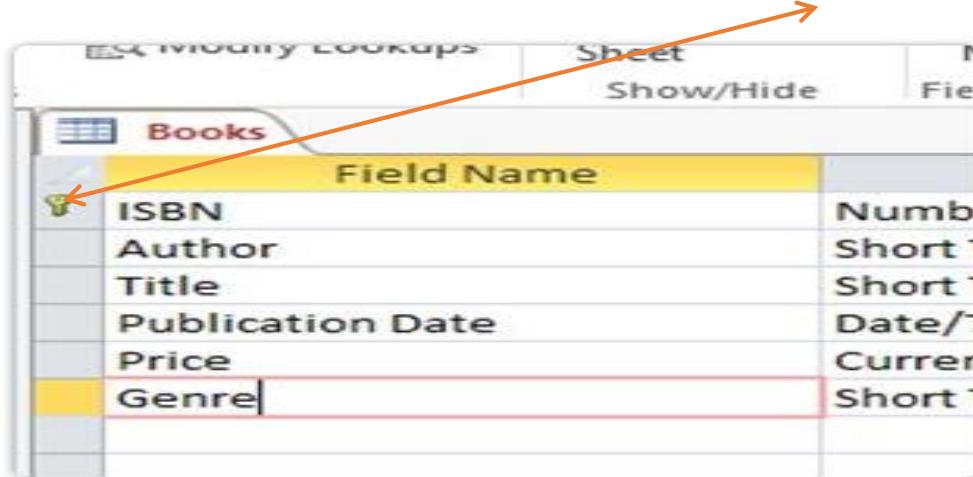
Next, we'll tab over to **Description** and **type a description** of the data entered in the field. To go back to the previous field at any stage, **hold Shift and press Tab**.

We can enter in information so that this table also has an **Author, Title, Publication Date, Price, and Genre** fields.

When we're finished entering all fields for this table, this is what we have:

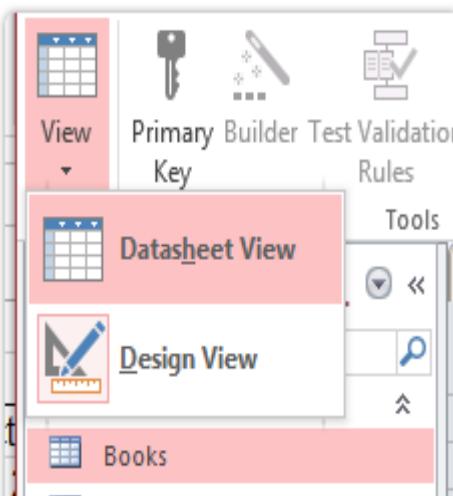
Field Name	Data Type	Description (Optional)
ISBN	Number	Library of Congress Catalog Number for Books
Author	Short Text	The author's name
Title	Short Text	The book's title
Publication Date	Date/Time	The copyright date listed inside the book
Price	Currency	The price of the book
Genre	Short Text	a genre of the book

Note the key beside ISBN. This is our **primary key**.

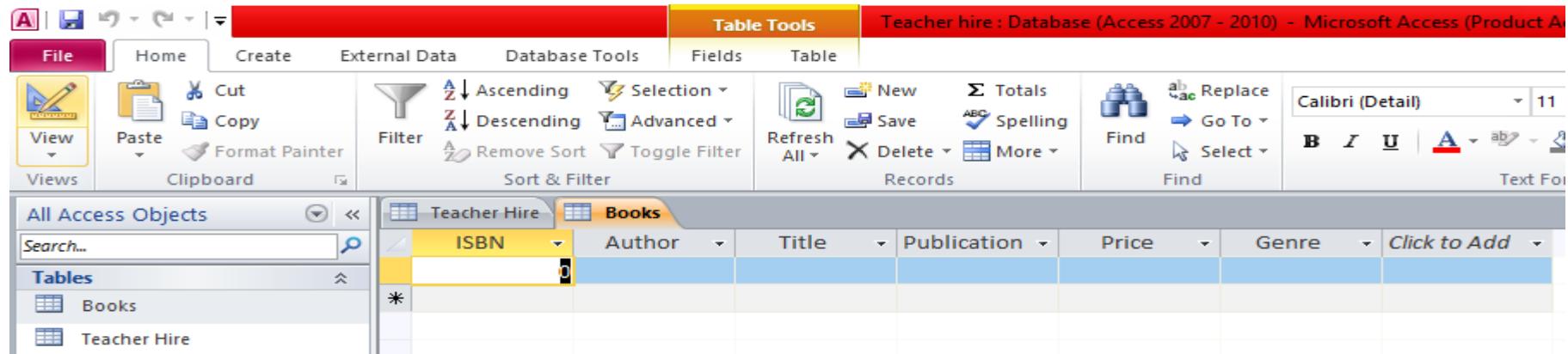


Primary Key		Field Name	Type
		ISBN	Number
		Author	Short Text
		Title	Short Text
		Publication Date	Date/Time
		Price	Currency
		Genre	Short Text

Once you're finished with this, change to the **Datasheet View**. You can do this by clicking on the dropdown arrow under the **View button** on the Design tab and click on **Datasheet View**



You'll be prompted to **save** the table, then you'll be able to enter information into it.
Click **Yes**.



Adding Data to a Table

A screenshot of the "Books" table in Datasheet view. A new row is being added at the bottom, indicated by a yellow asterisk (*) in the ISBN column and a cursor in the Click to Add column. The table structure is identical to the one in the previous screenshot, with columns: ISBN, Author, Title, Publication, Price, Genre, and Click to Add. The new row currently contains: ISBN *, Author (empty), Title (empty), Publication (empty), Price \$0.00, Genre (empty), and Click to Add.

ISBN	Author	Title	Publication	Price	Genre	Click to Add
62088025	James Grippando	Lying with Strangers	04/24/2012	\$8.99	Fiction	
316067369	Alice Sebold	The Almost Moon	09/08/2008	\$9.01	Fiction	
345531964	John Grisham	The Testament	12/27/2011	\$8.99	Fiction	
393059804	Sebastian Junger	A Death in Belmont	04/18/2006	\$9.58	Fiction	
440245117	John Grisham	The Confession	07/19/2011	\$6.04	Fiction	
618915478	Philip Roth	Exit Ghost	10/01/2007	\$7.37	Fiction	
*				\$0.00		

Now, we're going to create a second table, and then link it to the first table. This is the main benefit of creating databases. It allows you to create separate sets of data, and link them together, to keep your information organized.

We are going to create a simple Orders table. This table will store some data about orders that have been made for books.

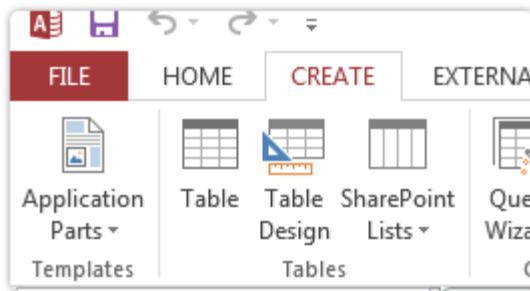
Traditionally, order tables can be quite complicated. Orders can have multiple items, items can belong to multiple orders, and orders can have different prices or discounts.

For our example, though, we're going to make it simple.

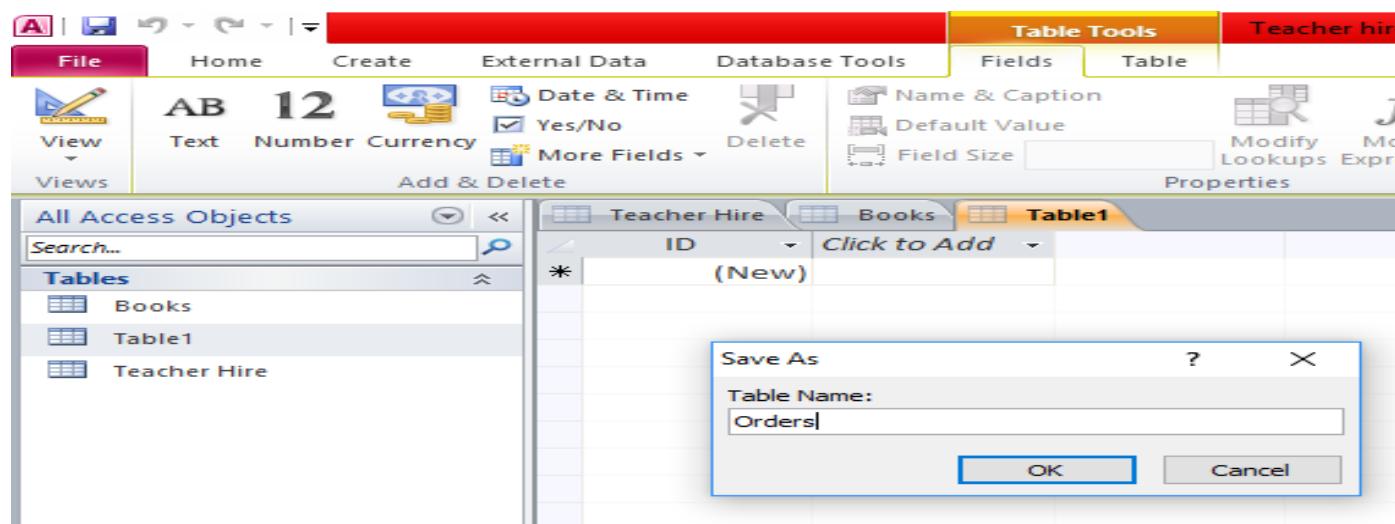
- One book per order
- No price changes
- Only some information about destinations

Creating a Third Table

Click on the **Create tab** and then on **Table**, just as you did earlier. A new table will be created.

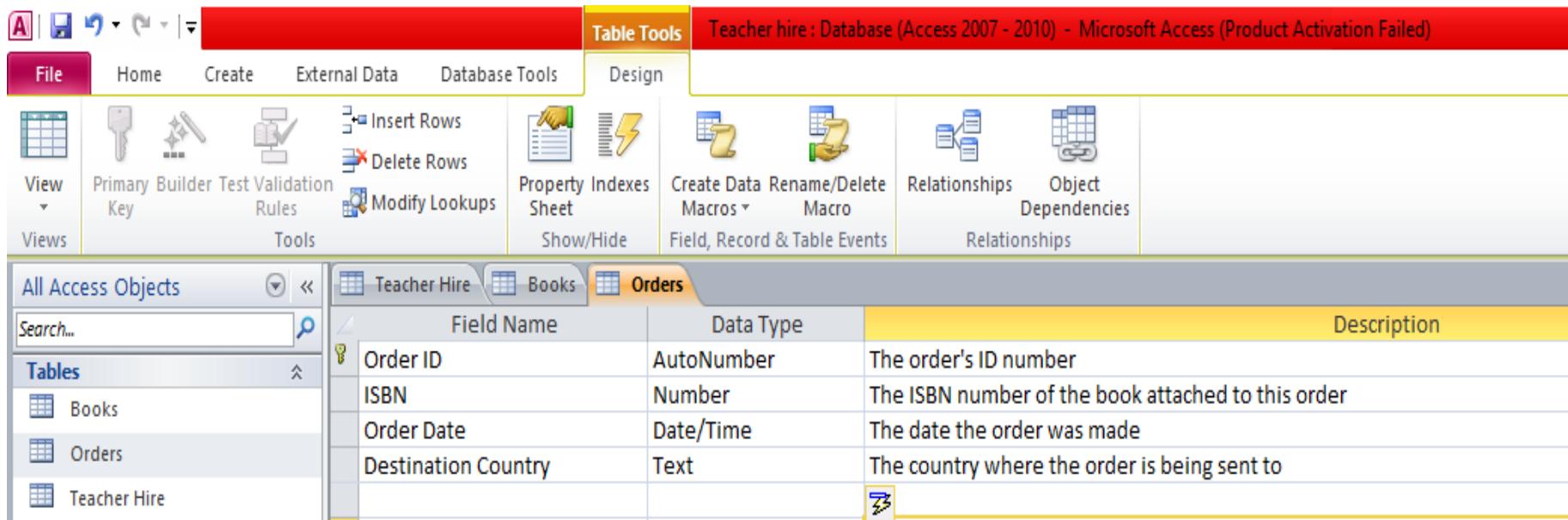


Click on the **View button** and select **Design View**. This is where we define the fields. Access will first ask you to save the table. Enter "**Orders**" and click **OK**.



Now, enter the following fields into the **Design view**:

Your Design View should look like this:

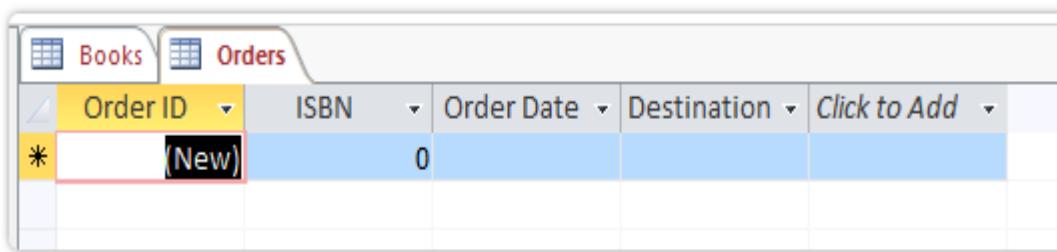


The screenshot shows the Microsoft Access ribbon with the 'Table Tools' section selected, specifically the 'Design' tab. On the left, there's a navigation pane titled 'All Access Objects' with a search bar and a list of tables: Books, Orders, and Teacher Hire. The 'Orders' table is currently selected. The main area displays the table structure with three columns: 'Field Name', 'Data Type', and 'Description'. The data is as follows:

Field Name	Data Type	Description
Order ID	AutoNumber	The order's ID number
ISBN	Number	The ISBN number of the book attached to this order
Order Date	Date/Time	The date the order was made
Destination Country	Text	The country where the order is being sent to

Save your database, and then click on the **View button** to change to **Datasheet view**.

Your table starts like this:

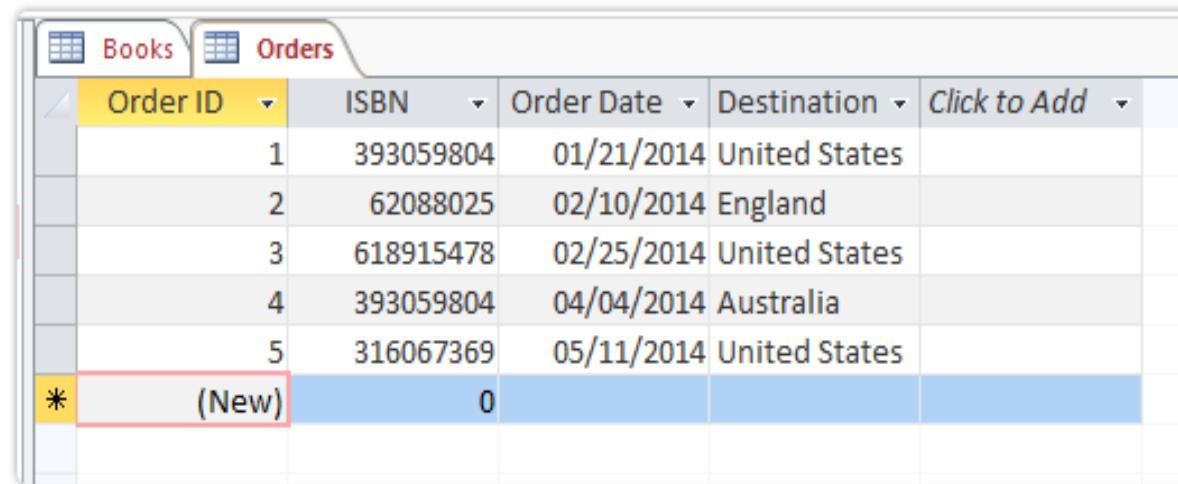


The screenshot shows the Microsoft Access 'Datasheet' view for the 'Orders' table. The table has four visible columns: 'Order ID', 'ISBN', 'Order Date', and 'Destination'. A fifth column, 'Click to Add', is present but empty. The first row is a new record, indicated by the asterisk (*) in the 'Order ID' cell and the '(New)' placeholder text. The 'ISBN' cell contains the value '0'.

Order ID	ISBN	Order Date	Destination	Click to Add
*	(New)	0		

Enter in these values for the orders. You can just press Tab when you select the Order ID field, as you don't need to type anything in

Your table should now look like this:



The screenshot shows a Microsoft Access table window titled "Orders". The table has five columns: "Order ID", "ISBN", "Order Date", "Destination", and "Click to Add". There are five data rows with Order IDs 1 through 5, and a new row at the bottom with an asterisk (*) in the Order ID column and "(New)" in the ISBN column. The "Click to Add" column contains a blue button.

Order ID	ISBN	Order Date	Destination	Click to Add
1	393059804	01/21/2014	United States	
2	62088025	02/10/2014	England	
3	618915478	02/25/2014	United States	
4	393059804	04/04/2014	Australia	
5	316067369	05/11/2014	United States	
*	(New)	0		

Linking Two Tables:

We have **two tables**: **Books**, and **Orders**. Now it's time to link them together.

To be able to link them together, we need a field in both tables that match. They don't need to be the same name, they just need to contain the same data.

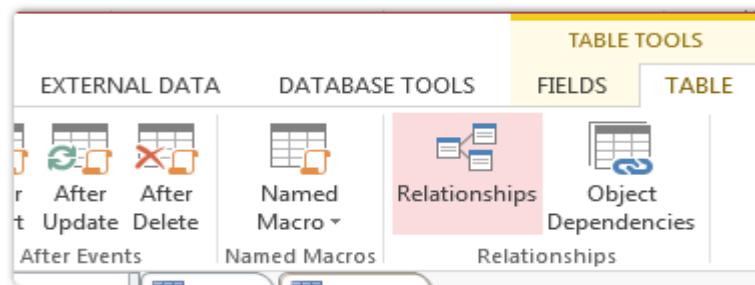
Normally, you would use some kind of **ID number**, whether it's a real number (social security number, phone number) or a made up number (**employee ID**, **author ID**).

This is good database design, as you want the field that links the tables (the **ID**) to be different to the label for those fields (book title, order destination, etc.), in case the labels change.

In this example, we'll be using the **ISBN** to link the **two tables**. Save the table, and then click on the **Table tab**.

Linking Two Tables

Click **Relationships.**



If you have no relationships created, this form will be displayed

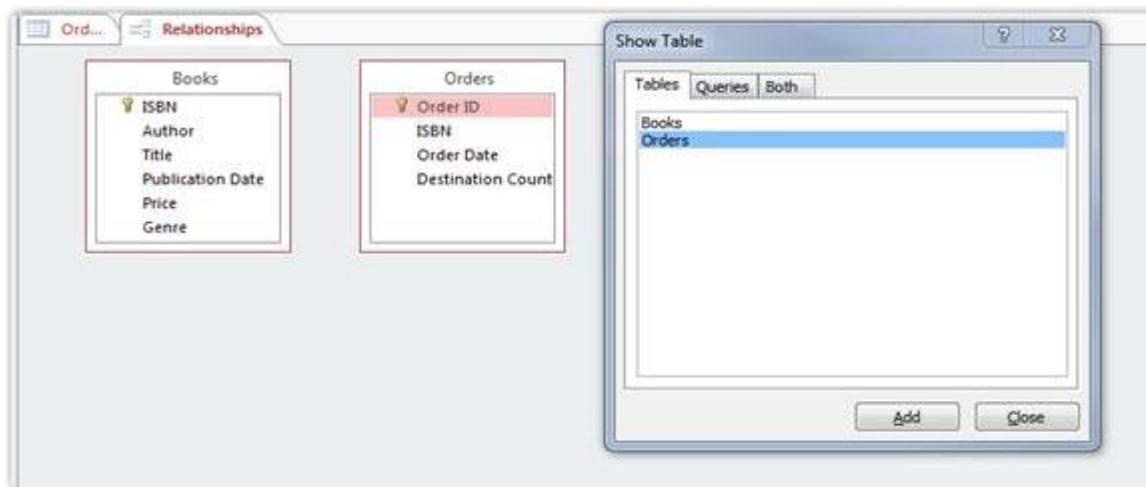
A screenshot of Microsoft Access showing the 'Relationship Tools' ribbon tab selected. On the left, there's a toolbar with icons for Edit Relationships, Clear Layout, Relationship Report, and Tools. The 'Tools' section shows 'All Access Objects' with a search bar and a list of Tables: Books, Orders, and Teacher Hire. In the center, there's a 'Relationships' ribbon tab with options like Hide Table, Direct Relationships, All Relationships, and Close. A 'Show Table' dialog box is open, listing Tables: Books, Orders, and Teacher Hire. The 'Tables' tab is selected. At the bottom right of the dialog are 'Add' and 'Close' buttons.

This view will show each of the tables you have and how they relate to each other. First, we need to specify what tables to show.

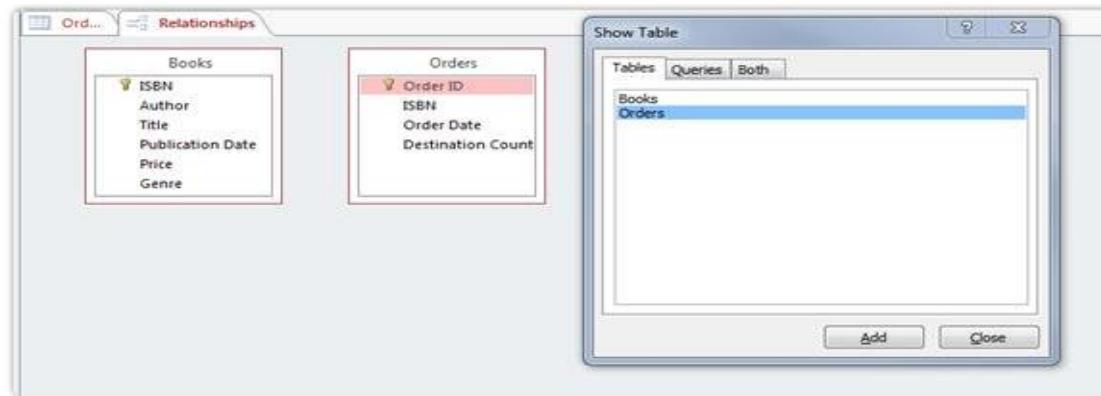
In this window (pictured above), you see all of the tables and queries available in your database. If you want to see just Tables, click the Tables tab, if you want to see just queries, click the Queries tab, but if you want to see both, click the Both tab.

Select the tables or queries you want to create relationships for, then click Add. You may have to press the CTRL key to be able to select more than one.

In this list, we have both tables. Click Add after selecting each table to add them to the Relationship View.

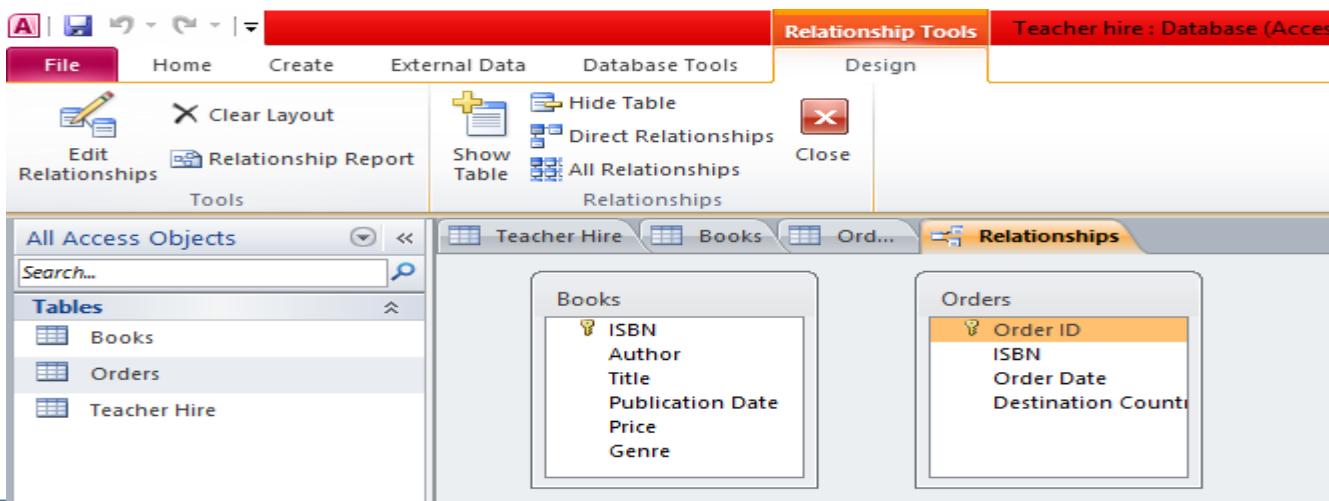


In this list, we have both tables. Click **Add** after selecting each table to add them to the Relationship View.

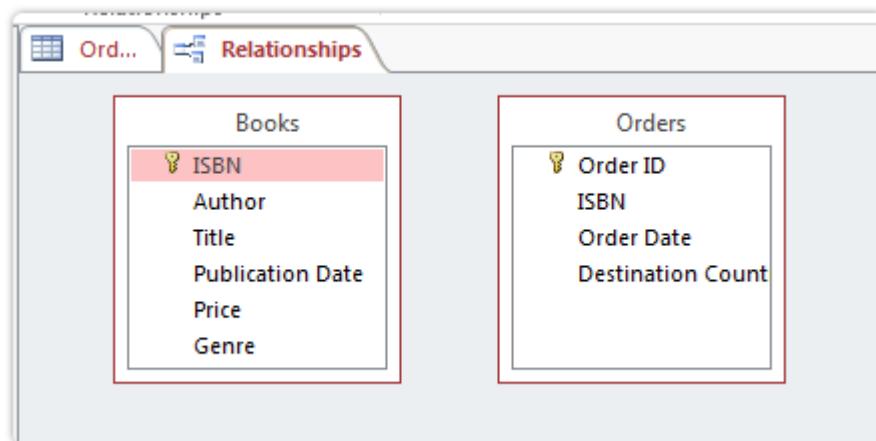


Now, click **Close**.

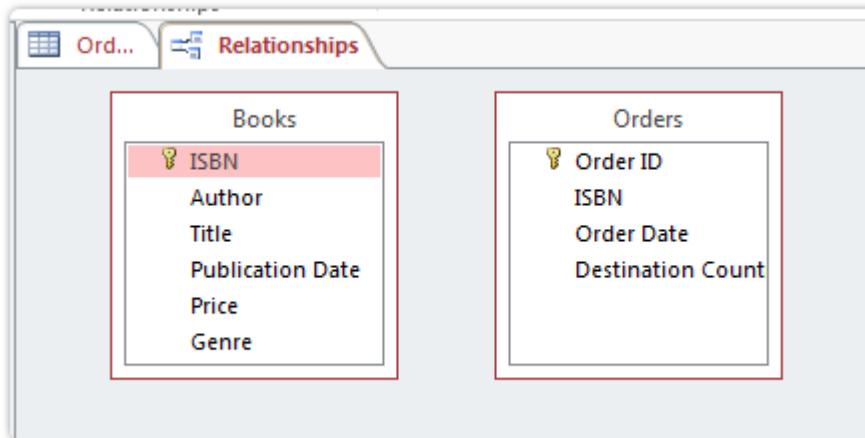
The Relationship View is now shown.



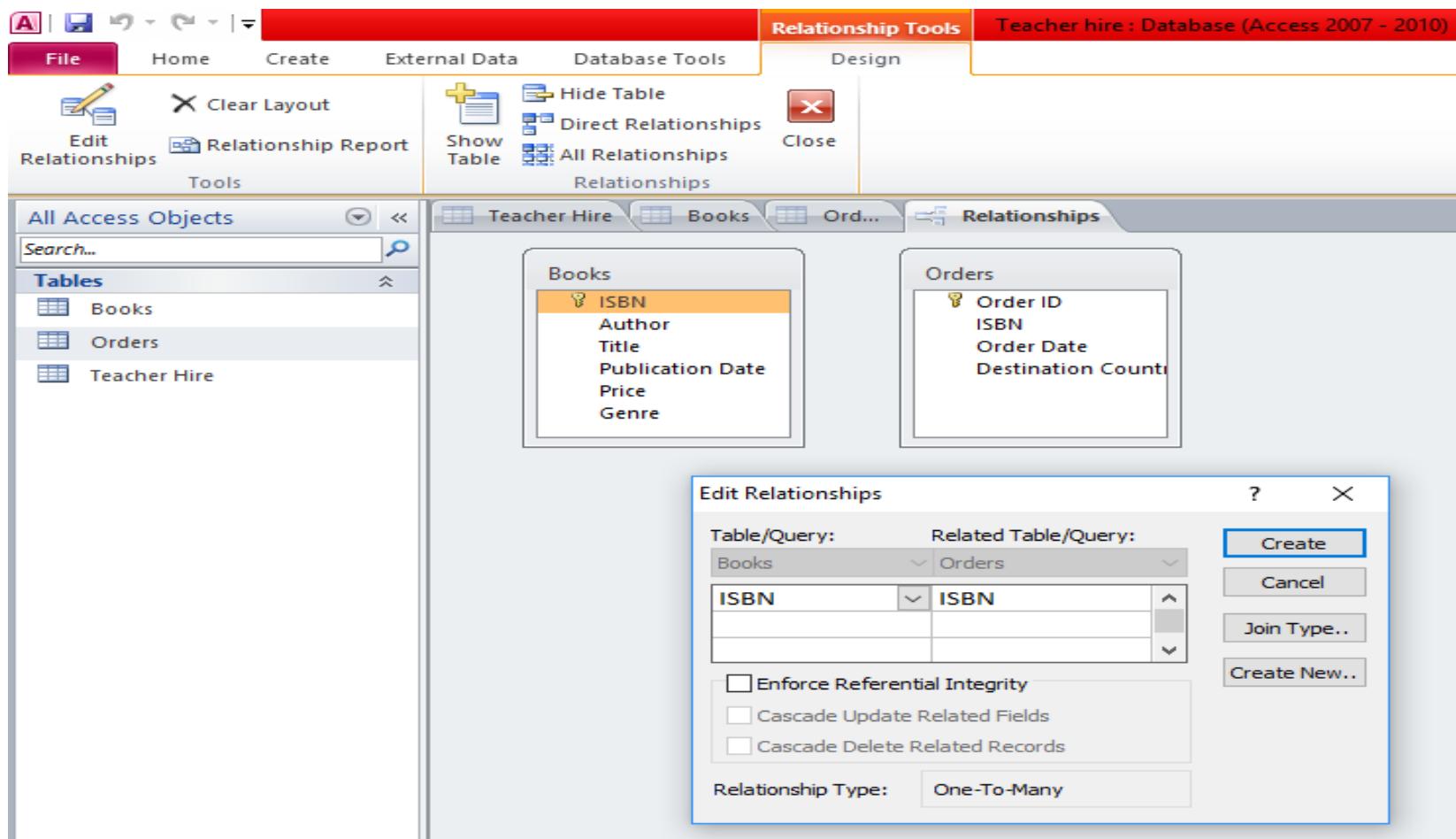
Click on the **ISBN** field in the **Books table**.



Click and drag the word **ISBN** in the **Books table** onto the word **ISBN** in the **Orders table**

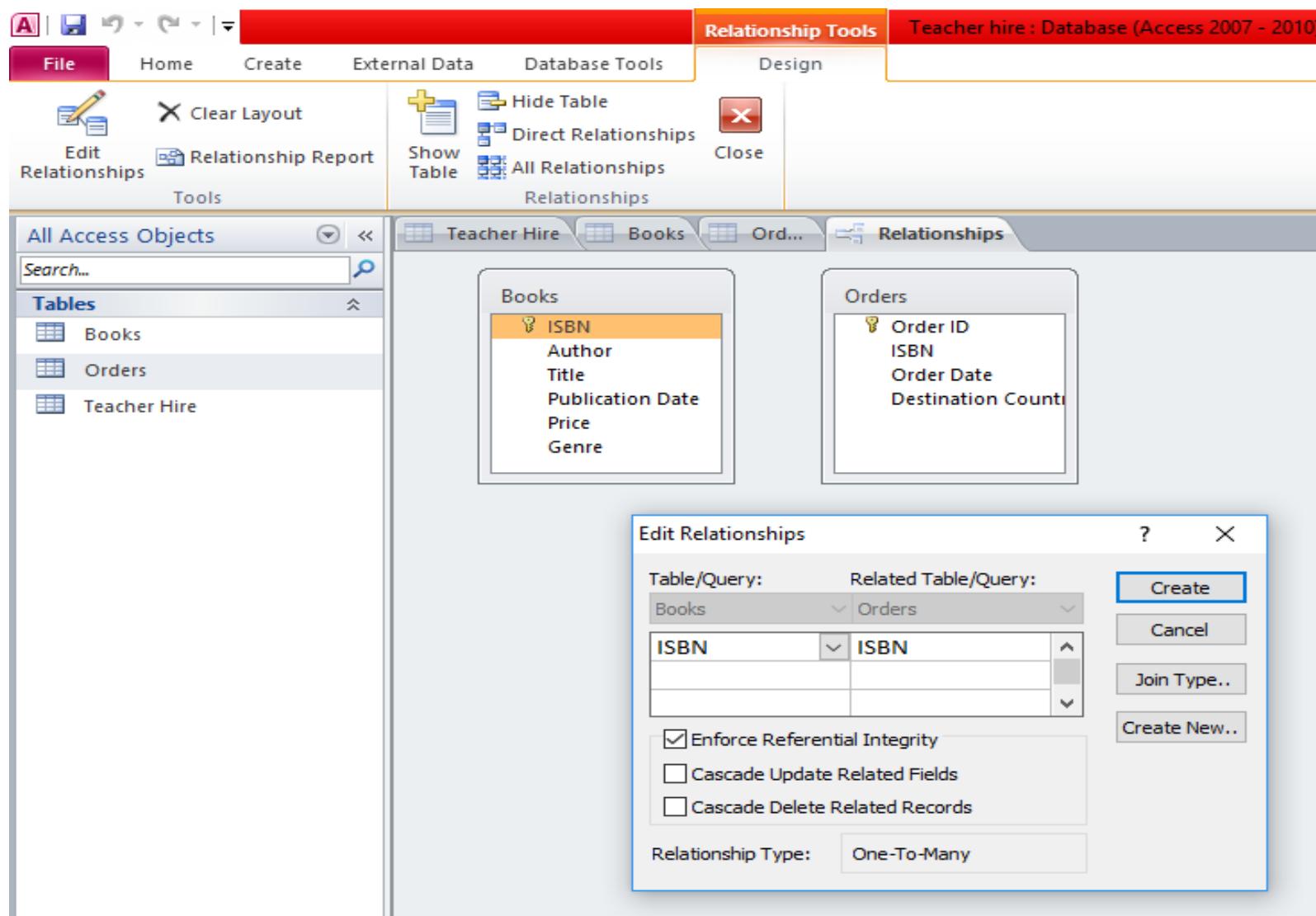


The **Edit Relationships** dialog box will appear.

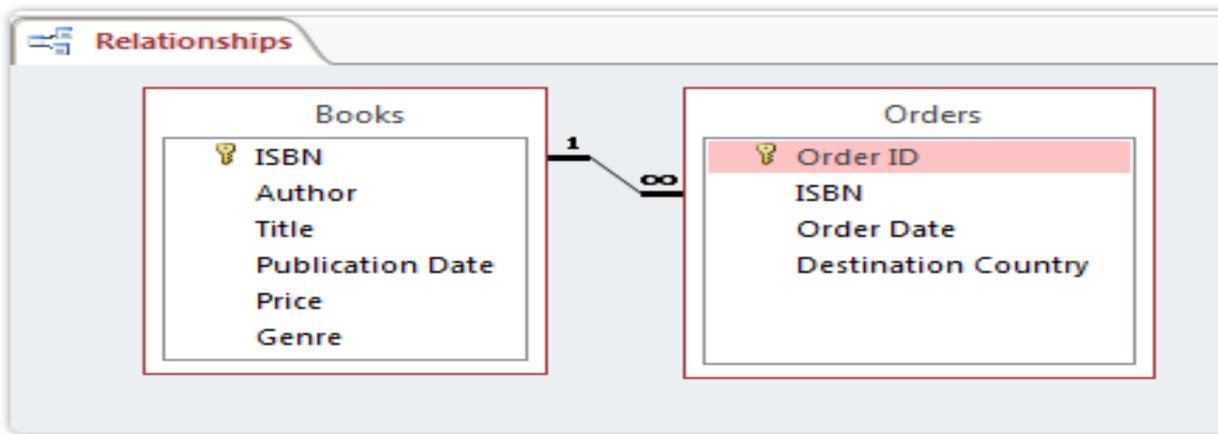


This means that the ISBN field in the Books table will be related to the ISBN field in the Orders table

Check **Enforce Referential Integrity** as shown below and Click **Create** when you're done.

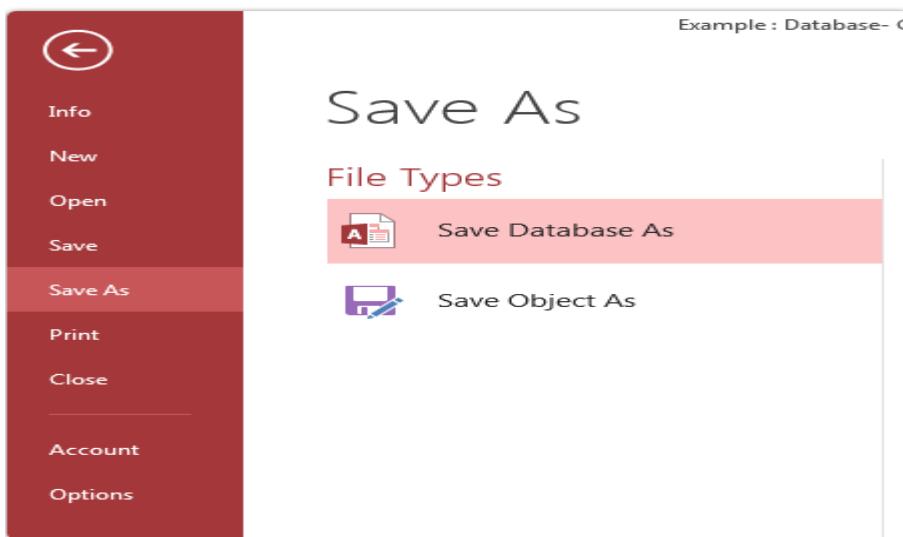


Once the relationship is created, the Relationship View will look like this:



Save a Database

To save a table, click Save Object As under the File tab.



Querying the Database

One of the main reasons that we use databases is that we can use them to **quickly find information**. In database terms, this search is called a **query**.

Query's in databases can be very simple e.g. **searching** for everybody with the **first name** "Noureddine".

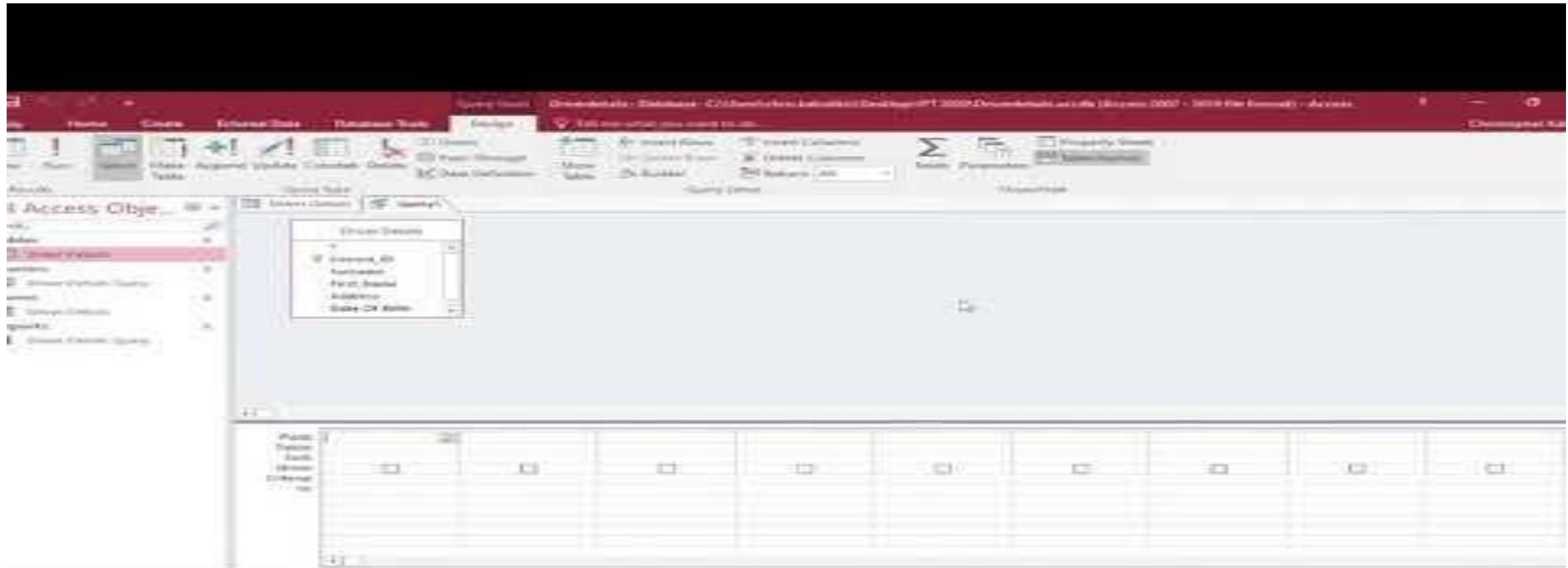
Watch video for Access: Designing a Simple Query:

https://www.youtube.com/watch?v=LUL1nnxUz_c



Watch video for Database Query By Example (QBE):

https://www.youtube.com/watch?v=aV3_74l4bpY



Search Operators: To help you write your query correctly there are a few set operators that have different meaning, these are detailed in the table below:

	Value	Value
<	Less than	Age < 10
<=	Less than or equal to	Age <= 10
>	Greater than	Height > 105cm
>=	Greater than or equal to	Height >= 105
=	Equal to	Cost = 100
<>	Between two values	Height 105 <> 160

These are a comprehensive set of operators which allow you to tailor your search perfectly. It is also possible to combine more than one search term / operator.

There are two **logical operators** which allow you do combine search criteria, these are **AND** and **OR**

AND = Will return records where all search terms are matched

OR = Will return records where at least one search term is matched:

Putting this in to practice

When using Microsoft Access to query a database the user interface will look like this.

Field:	Vehicle ID	Brand	Model	Doors	Mileage	Year of registration	Price
Table:	Cars						
Sort:							
Show:	<input checked="" type="checkbox"/>						
Criteria:							
or:							

You must then place your search criteria into the 'Criteria' Row of the correct field.

For example, If you wish to search for **all cars cheaper than \$5000** your search term would be **<5000** and you would place it in the **price field**, see below:

Field:	Vehicle ID	Brand	Model	Doors	Mileage	Year of registration	Price
Table:	Cars						
Sort:							
Show:	<input checked="" type="checkbox"/>						
Criteria:							<5000
or:							

Putting this in to practice

AND:

If you wished to **combine two search terms** where they must both be correct you will need to use the **AND** operator. In Microsoft Access you would achieve this simply by placing another search criteria in a fields criteria box.

So if for example you wanted to search for **all ford cars that cost more than \$16000**, the search term would be:

(Model = 'Ford') AND (Price>'16000')

In access this would look like:

Field:	Veichle ID	Brand	Model	Doors	Mileage	Year of registration	Price
Table:	Cars						
Sort:							
Show:	<input checked="" type="checkbox"/>						
Criteria:			"Ford"				>16000
or:							

Putting this in to practice

OR:

If you wished to **combine two search terms** where only one of them needs to be correct you will need to use the **OR** operator. In Microsoft Access you would achieve this simply by placing another search criteria in another fields criteria box, however this time you will use the row below labelled 'or'.

So if for example you wanted to search for all **Ford or Fiat** cars that were **less than \$10000**, the search term would be:

(Model = 'Ford' OR 'Fiat') AND (Price < 10000)

In Access this would look like:

Field:	Vehicle ID	Brand	Model	Doors	Mileage	Year registered	Price
Table:	Cars						
Sort:							
Show:	<input checked="" type="checkbox"/>						
Criteria:		"Ford"					<10000
or:		"Fiat"					

Exam Tips:

In the exam you may be asked to write a search criteria for a required situation e.g.
search for all **Ford or Fiat** cars that were **less than \$10000**

For this the answer will look like this - **(Model = 'Ford' OR 'Fiat') AND (Price < 10000)**
Be careful with your placement of brackets and choice of **AND v OR**

You may also be provided with a table that looks like the Microsoft Access query UI as seen above. Here you will be given a table to refer to and a query to set up.

Key points:

- Read the question carefully as you may not be required to include every field.
- If you are required to search using a specific field but NOT required to show it, you must not tick the "**Show**" box

Practical Activity on Database Query :

1. Create a folder in your desktop to save all your database files.
2. Open Access 2013
3. Open the step by step Practical Activity on Database Query on this slide or you access it using Edmodo or TEAMS .
4. You have to import the csv file called **Football.csv** and **Movie.csv** from your Edmodo or on TEAMS



Adobe Acrobat
Document



Microsoft Excel
97-2003 Worksheet



Microsoft Excel
97-2003 Worksheet

● Key definitions



Term	Definition
Database	A structured collection of data
Table	Contains data about one type of item or person or event, using rows and columns
Record	A row within a table that contains data about a single item, person or event
Field	A column which contains one specific piece of information
Primary key	A unique reference for each record

Record 1	Primary key	Field 1	Field 2	Field 3	Field 4
Record 2	Primary key	Field 1	Field 2	Field 3	Field 4
Record 3	Primary key	Field 1	Field 2	Field 3	Field 4
Record 4	Primary key	Field 1	Field 2	Field 3	Field 4
Record 5	Primary key	Field 1	Field 2	Field 3	Field 4
Record 6	Primary key	Field 1	Field 2	Field 3	Field 4

Learning Objectives

• 8.1 Database Concepts

- Show understanding of the limitations of using a file-based approach for the storage and retrieval of data
- Describe the features of a relational database that address the limitations of a file-based approach
- Show understanding of and use the terminology associated with a relational database model:
Including entity, table, record, field, tuple, attribute, primary key, candidate key, secondary key, foreign key, relationship (one-to-many, one-to-one, many-to-many), referential integrity, indexing
- Use an entity-relationship (E-R) diagram to document a database design
- Show understanding of the normalization process: First Normal Form(1NF), Second Normal Form (2NF) and Third Normal Form (3NF)
- Explain why a given set of database tables are, or are not, in 3NF
- Produce a normalized database design for a description of a database, a given set of data, or a given set of tables

Learning Objectives

- **8.2 Database Management System (DBMS)**

- Show understanding of the features provided by a Database Management System (DBMS) that address the issues of a file based approach Including:
 - ❖ data management, including maintaining a data dictionary
 - ❖ data modelling
 - ❖ logical schema
 - ❖ data integrity
 - ❖ data security, including backup procedures and the use of access rights to individuals / groups of users
- Show understanding of how software tools found within a DBMS are used in practice, Including the use and purpose of:
 - ❖ developer interface
 - ❖ query processor

Relational databases

Relational databases allow data to be separated and connected across several tables. Tables are connected through primary and foreign keys to increase efficiency.

A single **flat-file** table (mean: A database consisting of only one table often stored as a CSV file). is useful for recording a limited amount of data. But a large flat-file database can be inefficient as it takes up more space and **memory** than a relational database. It also requires new data to be added every time you enter a new record, whereas a relational database does not. Finally, **data redundancy** – where data is partially duplicated across records – can occur in flat-file tables, and can more easily be avoided in relational databases.

Therefore, if you have a large set of data about many different **entities**, it is more efficient to create separate tables and connect them with relationships.

Relational databases allow data to be stored in a clear, organised manner across multiple tables. Links, known as relationships, are formed to allow the data to be shared across the tables.

For instance, a retail company might have different tables for the following information:

- customer details
- customer orders
- product details
- stock levels
- stock locations
- staff details

Relational databases

Product details could be complicated, example: if the company sold books there may be several categories within books, including author name, title, genre, physical size and many other details. Storing all this information in one flat-file table would create a very large table.

Normalisation:

Normalisation is the process of analysing how to make databases more efficient by using separate tables to reduce **redundant data**. When a database is normalised, data is broken down into smaller tables and relationships are used to link them.

Connecting entities:

The main characteristics of a **relational database** are:

it is built from a set of unique tables (also called relations)

a table contains data about just one **entity**

tables must have a **primary key**

tables are linked by primary and **foreign keys**

When working with relational databases, users need to try to keep **information about different entities in separate tables**. Each entity has a primary key to provide a unique reference to an entity, which means that an entity can be referenced in another table without having to call up all the details about that entity.

Relational databases

Entity relationship diagrams:

Entities can relate to each other in three different ways: **one to one**, **one to many** and **many to many**.

You can represent these relationships using an entity relationship diagram (ERD).

One to one

For example, **one** person has **one** address.



One to many

For example, **one** cinema has **many** customers.



Many to many

For example, **many** subjects can be taken by **many** students.

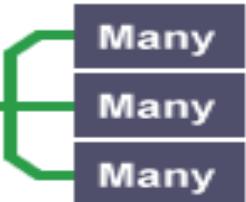


Photo courtesy by BBC Bitesize

Relational databases

Relationship example:

The following example shows how tables can be connected using **primary** and **foreign keys**.

The tables are in a **database** for an online shop. There are three tables:

- customer
- product
- Orders

Each table has a primary key field and each record has a primary key with a unique number.

Customer table:

The customer table gives customers a unique Customer ID (the primary key for this table) and shows customer details, ie name, address and phone number.

Customer ID	First name	Surname	Address	Phone number
02942	Rebecca	Johnson	49 Drew Road	029 381834
02943	Mushtaq	Aqbar	28 Lyttleton Lane	028 282738

Photo courtesy by BBC Bitesize

Relational databases

Product table:

The product table gives details about the products. The Product ID is the primary key for this table

Product ID	Product type	Colour	Size	Cost
284758	Jeans	Blue	28	£14.99
384957	Shoes	Brown	6	£12.99
483927	Jumper	Red	M	£29.99
489320	Shirt	Blue	M	£33.99
839258	Socks	White	6	£10.00

Orders table:

In the orders table, each order has a unique Order number (the primary key for this table). The table also includes customer ID (the primary key of the customer table) and product ID (the primary key of the product table) as **foreign keys**, but does not need to include all details about customers and products as these are stored in the Customer and Product tables.

Order number	Customer ID	Product ID	Quantity	Total cost
59876	02942	284758	2	£29.98
59877	02942	384957	3	£38.97
59878	02942	483927	1	£29.99
59879	02943	489320	3	£101.97
59880	02943	839258	2	£20.00

Relational databases

Normalisation

If the Orders table did not use Customer ID and Product ID fields, it would need to include additional fields from the Customer table and the Product table – an extra eight fields. It would also need to repeat the same customer details for each order. The Orders table would be much larger, use more data, and be repetitive.

This table shows how the orders table would look **without** normalisation:

Order number	First name	Surname	Address	Phone number	Product type	Colour	Size	Cost	Quantity	Total cost
59876	Rebecca	Johnson	49 Drew Road	029 381834	Jeans	Blue	28	£14.99	2	£29.98
59877	Rebecca	Johnson	49 Drew Road	029 381834	Shoes	Brown	6	£12.99	3	£38.97
59878	Rebecca	Johnson	49 Drew Road	029 381834	Jumper	Red	M	£29.99	1	£29.99
59879	Mushtaq	Aqbar	28 Lyttleton Lane	028 282738	Shirt	Blue	M	£33.99	3	£101.97
59880	Mushtaq	Aqbar	28 Lyttleton Lane	028 282738	Socks	White	6	£10.00	2	£20.00

Normalising the data creates a selection of simpler tables which takes up less data.

Relational databases

Database tools:

Databases store data, but you also need to be able to search and filter the data to find and present results.

Some of the tools you use when working with databases include:

- forms
- **queries**
- reports
- modules

Forms:

Forms are used to enter data into a database. This is only required if the database needs a user to enter data. The form should make it clear what data should be inputted, example: if you are selling something, you would fill in a form which prompts you for information like product name, brand and size.

Some databases are filled with data without direct user interaction, example: records of sensor readings from a weather station.

Queries:

Queries are used to search and filter a database. For example, when shopping online you select the options that you need and run a search. You can also filter and sort results, example: you may want the cheapest item first, or the item with the least time left at auction. All these actions are searching, sorting and filtering – and they are all queries.

Relational databases

Database tools:

Reports:

Reports are used to **export** data and present it in a way that is easy to read. For example, your address book database is full of details such as addresses, emails, dates of birth, but you might want to run a report to present just names and phone numbers.

Modules:

Database **software** and languages contain **modules** - pre-written programs. However, when making a database you might think of actions you want to do that do not have a specific module. In this case, you can edit modules in the programming language and your own procedures.

Queries and SQL:

Queries

Databases allow us to store and filter data to find specific information. A database can be **queried** using a variety of methods, although this depends on the **software** you are using.

Databases can use query languages or graphical methods to interrogate the data.

Query language:

Query language is a **written language used only to write specific queries**. This is a powerful tool as the user can define precisely what is required in a database. **SQL** is a popular query language used with many databases.

Query by example (QBE):

QBE allows the user to **create queries based on a template**, usually a set of filters presented in a graphical form. If you are using database software it might have an option to connect blocks and set the filters you want. The system presents a blank record and lets you specify the **fields** and values that define the query.

Database management software like **MySQL**, **Microsoft Access** and **Oracle** have front-end graphical **interfaces** which make it easier to run QBE queries.

Queries and SQL:

Boolean operators:

In a **database** we often need to filter the data to group certain results. **Boolean operators** are used to filter databases using **AND**, **OR** or **NOT**. They can search multiple **fields** at the same time to help us retrieve the data that we need. They are used because they provide results which are 'true' or 'false'.

Search engines also make use of Boolean operators to filter results.

AND is used to search records that contain one piece of information AND another.

For example, a database of clothing could be full of different types of items. Each item of clothing has lots of attributes including price, brand, colour, quantity and material.

You might want to search for brown shoes. A query for the words *brown AND shoes* would return results that contain the words brown and shoes.

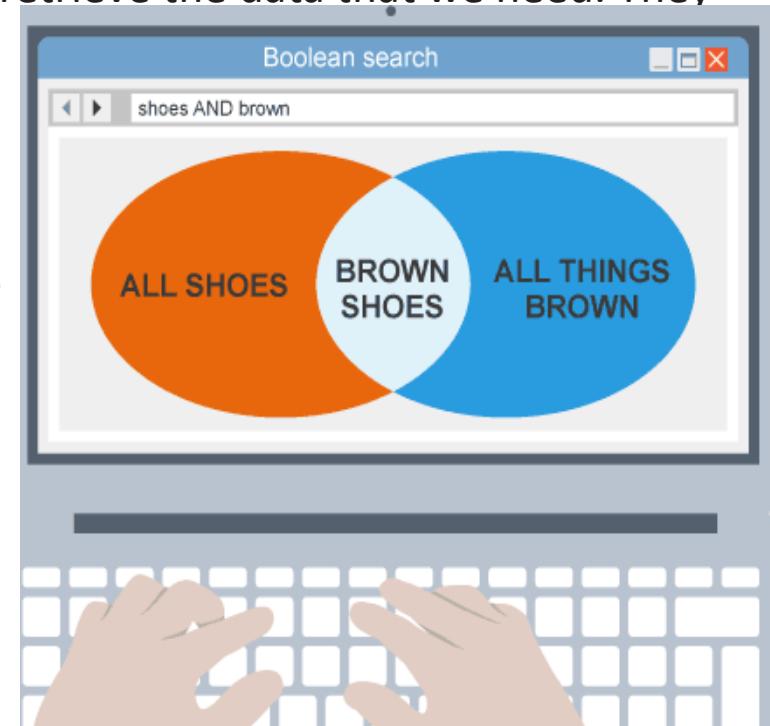


Photo courtesy by BBC Bitesize

In general, search engines treat the query *brown shoes* as *brown AND shoes*, which means that all results will contain both words, example: *brown trousers and red shoes for sale*.

Queries and SQL:

Boolean operators:

OR:

OR is used to search for records that contain EITHER one piece of information OR another.

It is used to request an alternative, for example *black shoes OR white shoes*. This would present results for **any shoes that were black or white**.

Most search engines use the OR function best if the search statements are defined by speech marks, eg "*brown shoes*" *OR "black jeans"*" would show pages which either contain brown shoes or black jeans.

NOT

NOT is used to exclude results.

The query *shoes NOT brown* will return results that **contain the word shoes but NOT the word brown**.

Some search engines use a minus sign in front of the word, instead of NOT, eg *-brown*. In these cases, if you run a search for *Doctor Who Capaldi* this will include all results with all of the words *Doctor*, *Who* and *Capaldi*, but the search phrase *Doctor Who –Capaldi* return the same results, excluding all results for *Capaldi*.

Queries and SQL:

Arithmetic operators:

A query can also be performed using **arithmetic operators**. These help to make specific searches related to numerical data.

Functions of arithmetic operators

This table shows some arithmetic operators and their functions:

Operator	Meaning
=	Equals
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

Queries and SQL:

Arithmetic operators:

Using arithmetic operators:

The table below shows some BBC TV programme listings:

ID	Title	Genre	Duration (mins)	Channel
01	EastEnders	Drama	30	BBC1
02	Dragons' Den	Entertainment	60	BBC2
03	The Voice	Entertainment	75	BBC3

Queries are useful for searching for specific conditions. You might want to find entertainment programmes on BBC3. A query for these conditions would look like this:

```
SELECT * FROM Programmes  
WHERE Genre='Entertainment' AND Channel='BBC3';
```

This would return the programme 'The Voice'.

You may want a programme that is less than 30 minutes long or is a nature programme.

A query for these conditions would look like this:

```
SELECT * FROM Programme  
WHERE Duration<30 OR Genre='Drama';  
This would return the programme "EastEnders"
```

SQL:

SQL is a programming language used to search and query **databases**. SQL gives you the ability to customise your queries. It is often used within database programs.

Uses of SQL

SQLite and **MySQL** are popular **open source** database **applications**. If you use a **blog** site, it is very likely that MySQL has been running behind the scenes to store entries as records and allow you to add, edit and delete blog posts.

SQL databases can be used to create lots of applications for use on the internet. They are often used by large companies because they allow the data **tables** to be stored on secure **servers**. A SQL server simply stores the data for a database - it does not provide **front-end** features. Therefore, the user does not need to know that a database is working behind a website.

A SQL database is manipulated using SQL code. SQL has also been designed so that lots of users can access it at the same time - it has a high capacity for storage space.

Syntax

SQL is an important programming language and understanding how to use it is a very important skill. The statements used in SQL code are quite similar to the words that you would normally use to find information. Because the statements are quite similar, it makes it easier to remember SQL **syntax**.

For example: "**SELECT** these fields **FROM** this table **WHERE** this is happening"

SQL:

For example: "**SELECT** these fields **FROM** this table **WHERE** this is happening"

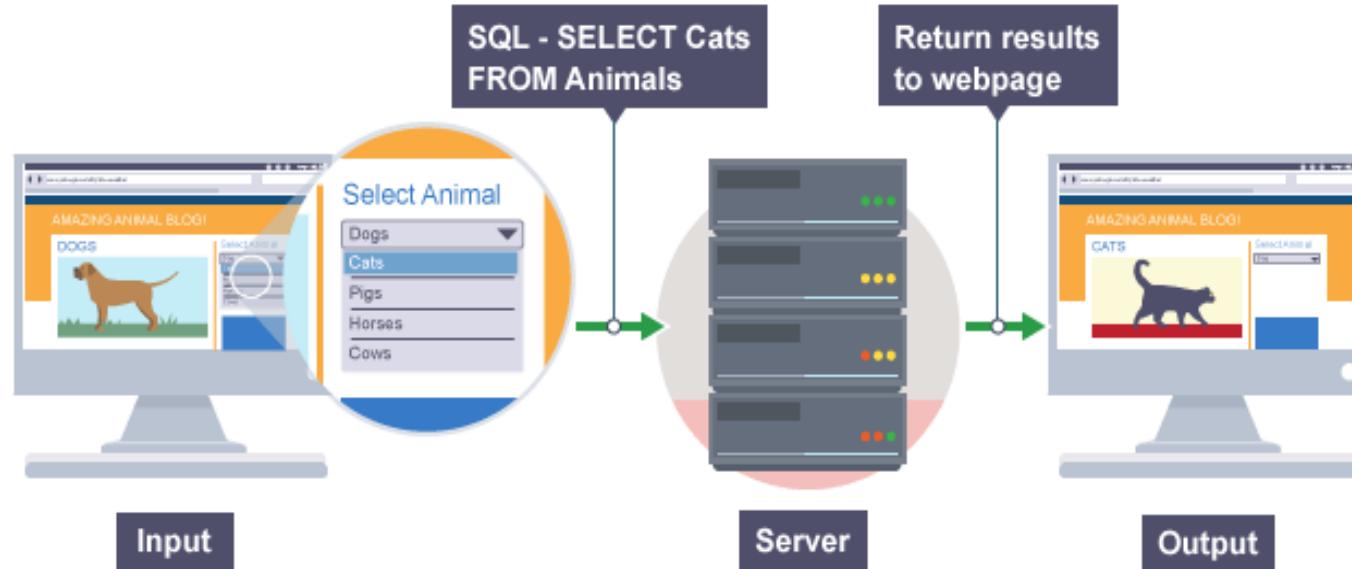


Photo courtesy by BBC Bitesize

Introduction to SQL

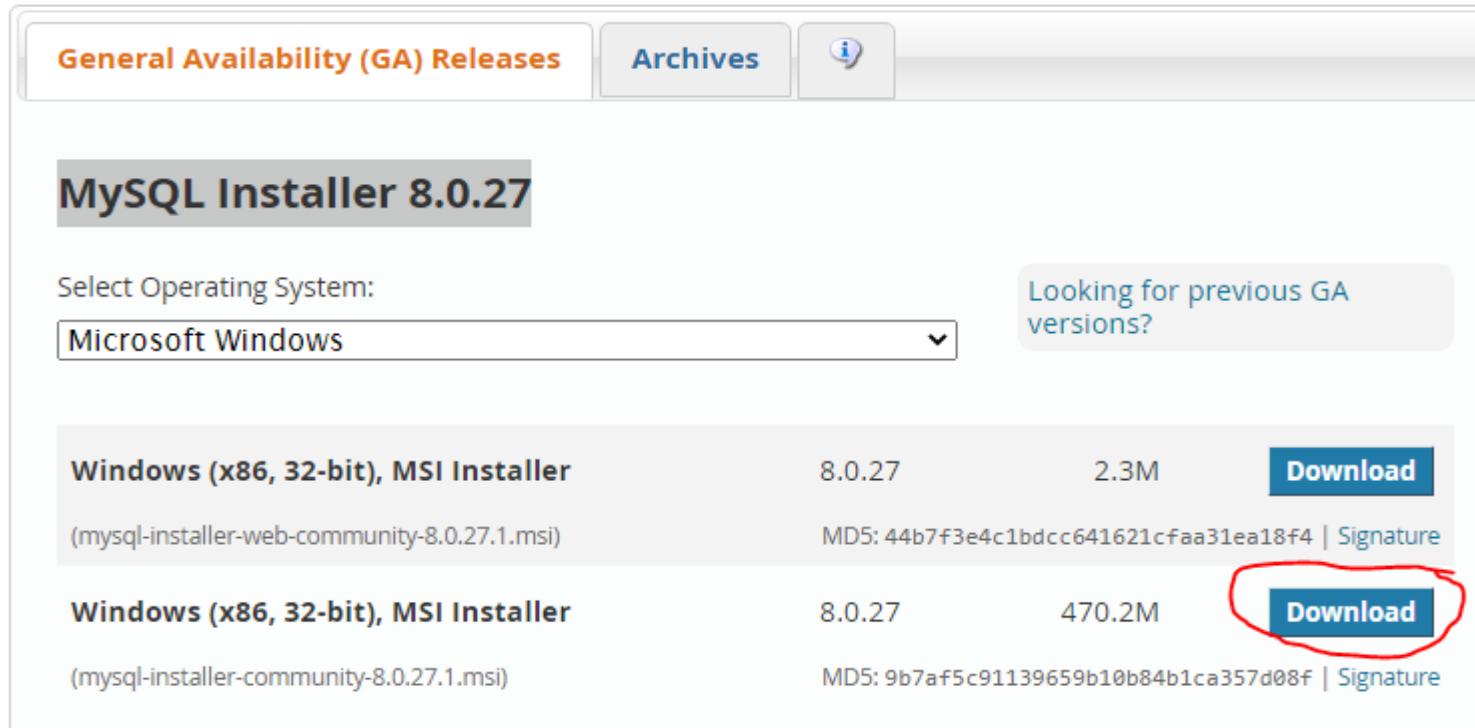
Click the link below to watch the video at home for SQL Tutorial - Full Database Course for Beginners? Duration; 4hours and 20 minutes

<https://www.youtube.com/watch?v=HXV3zeQKqGY>



Step by Step how to download MySQL server 8.0.27

Click the link to download MySQL Installer 8.0.27: <https://dev.mysql.com/downloads/installer/>



The screenshot shows the MySQL Installer 8.0.27 download page. At the top, there are tabs for "General Availability (GA) Releases" (highlighted in orange), "Archives", and a help icon. Below the tabs, it says "MySQL Installer 8.0.27". A dropdown menu "Select Operating System:" is set to "Microsoft Windows". To the right, a link says "Looking for previous GA versions?". There are two download options listed:

Installer Type	Version	File Size	Action
Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.27.1.msi)	8.0.27	2.3M	Download
Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.27.1.msi)	8.0.27	470.2M	Download

Please click link below to watch video How to install MySQL 8.0.27 Server and Workbench latest version on Windows 10

<https://www.youtube.com/watch?v=uKwR9fWsZJ4>

<https://www.youtube.com/watch?v=H0ZpCVhS6hw>

Step by Step how to download MySQL server 8.0.27

Click the link to download how to install MySQL 8.0.27 Server and Workbench :

<https://www.youtube.com/watch?v=H0ZpCVhS6hw>

Install MySQL 8.0.27
on Windows 10



Step by Step how to download MySQL server 8.0.27

Click the link to download how to create a new database and table with MySQL Workbench :

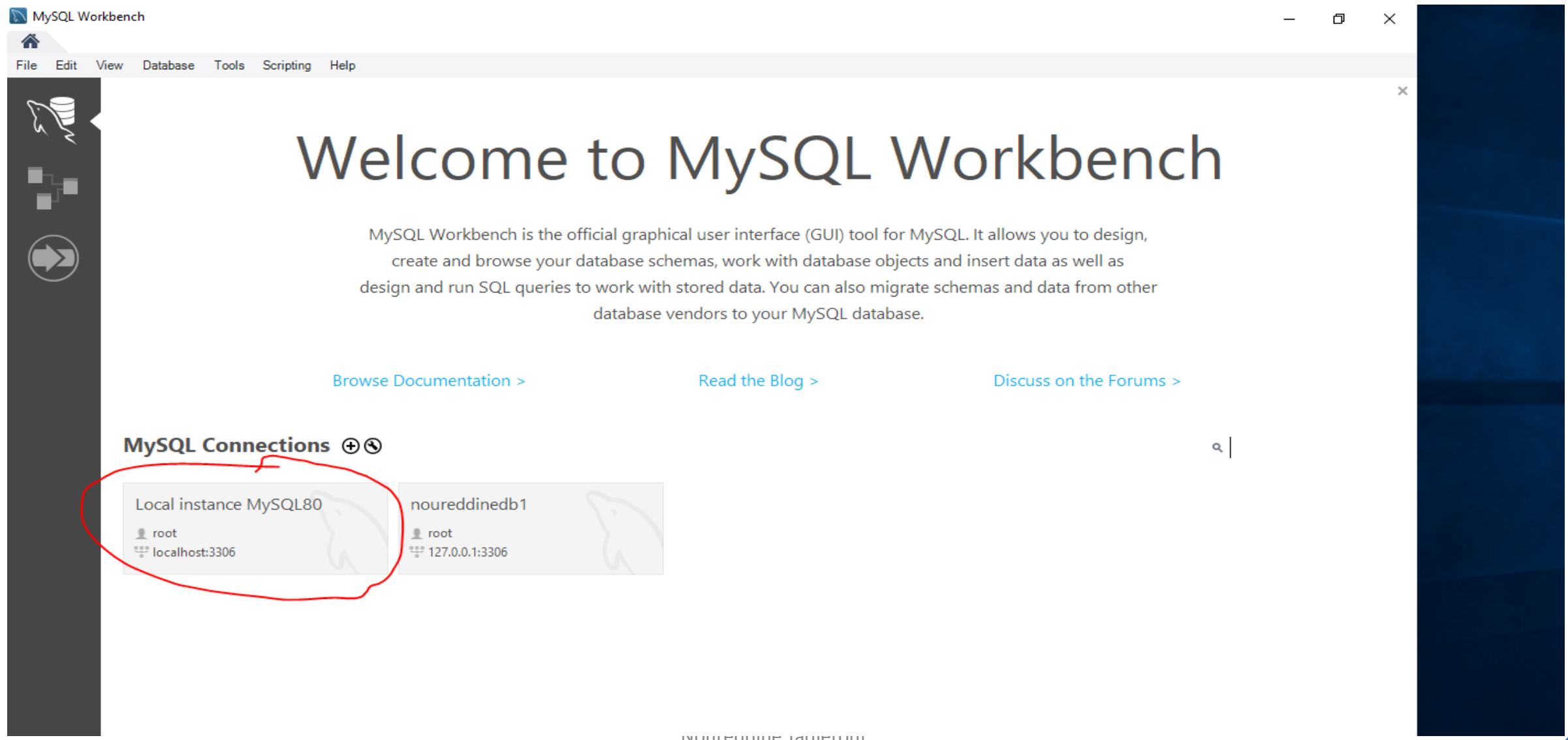
<https://youtu.be/7GbVRm9qo3U>

Create new database and table with MySQL Workbench 8.0.27



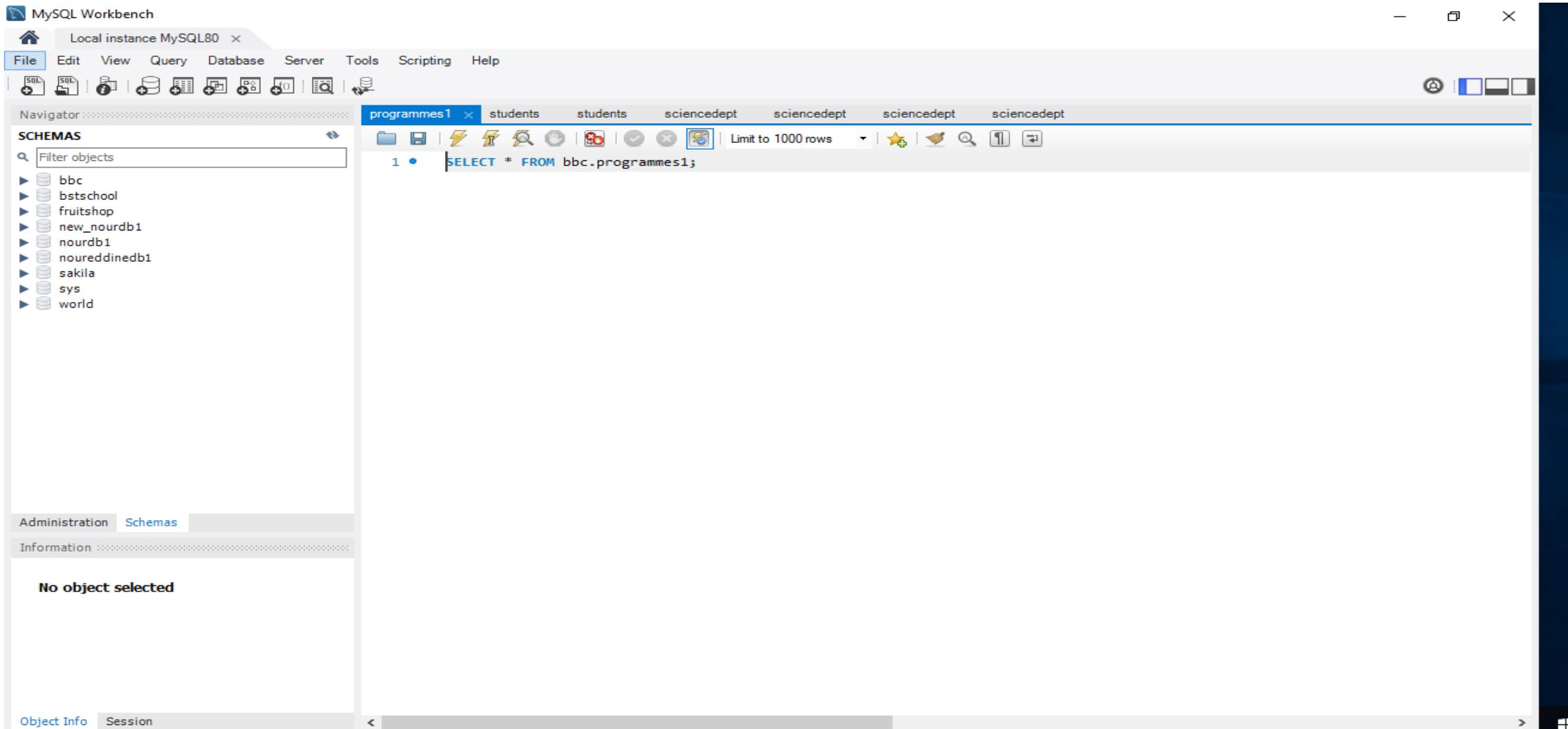
Using MySQL Workbench: Getting starter

Click Local instance MySQL 8.0



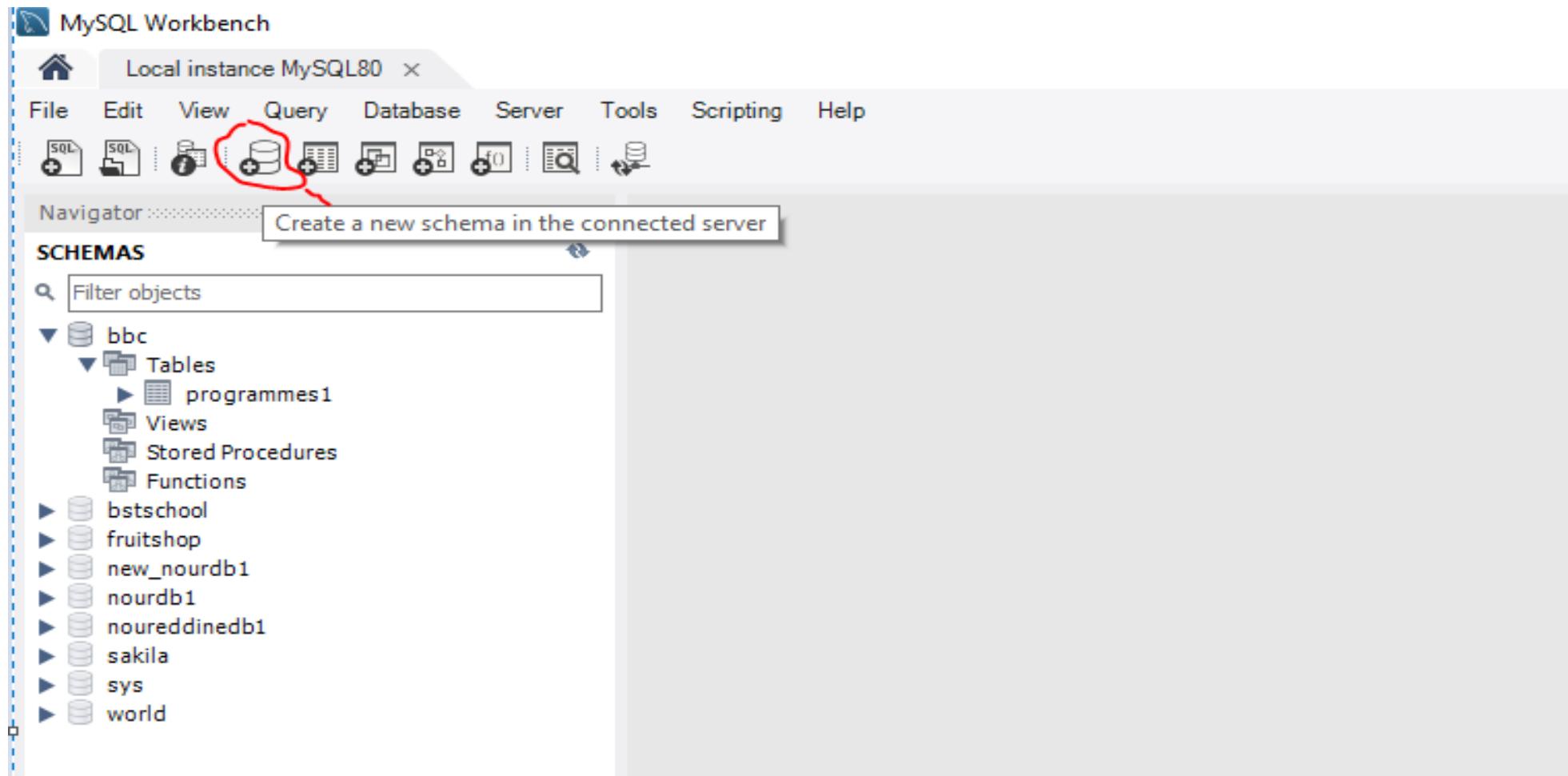
Using MySQL Workbench: Getting starter

Click Local instance MySQL 8.0



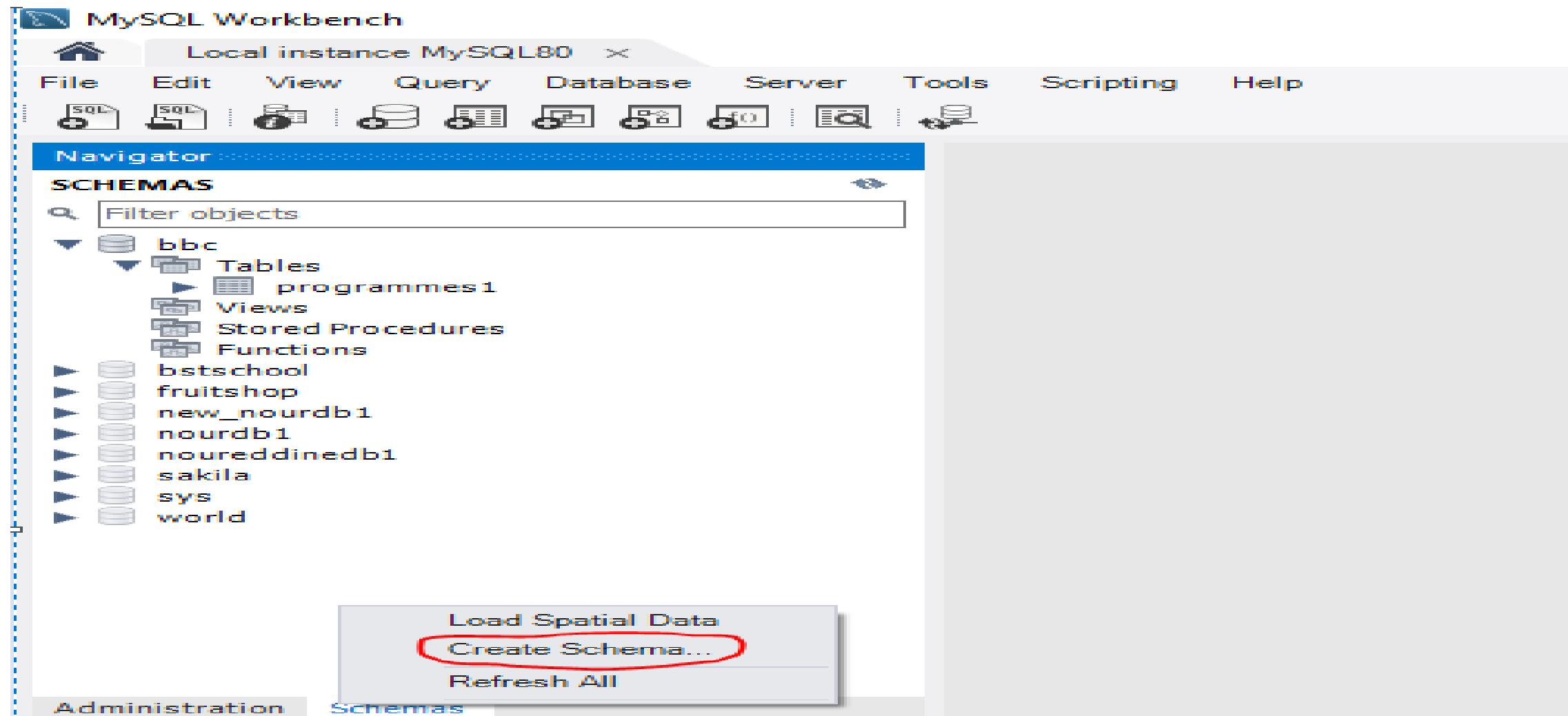
Using MySQL Workbench: Create a new schema it mean a new database

Click create a new schema in the connected server as shown below:



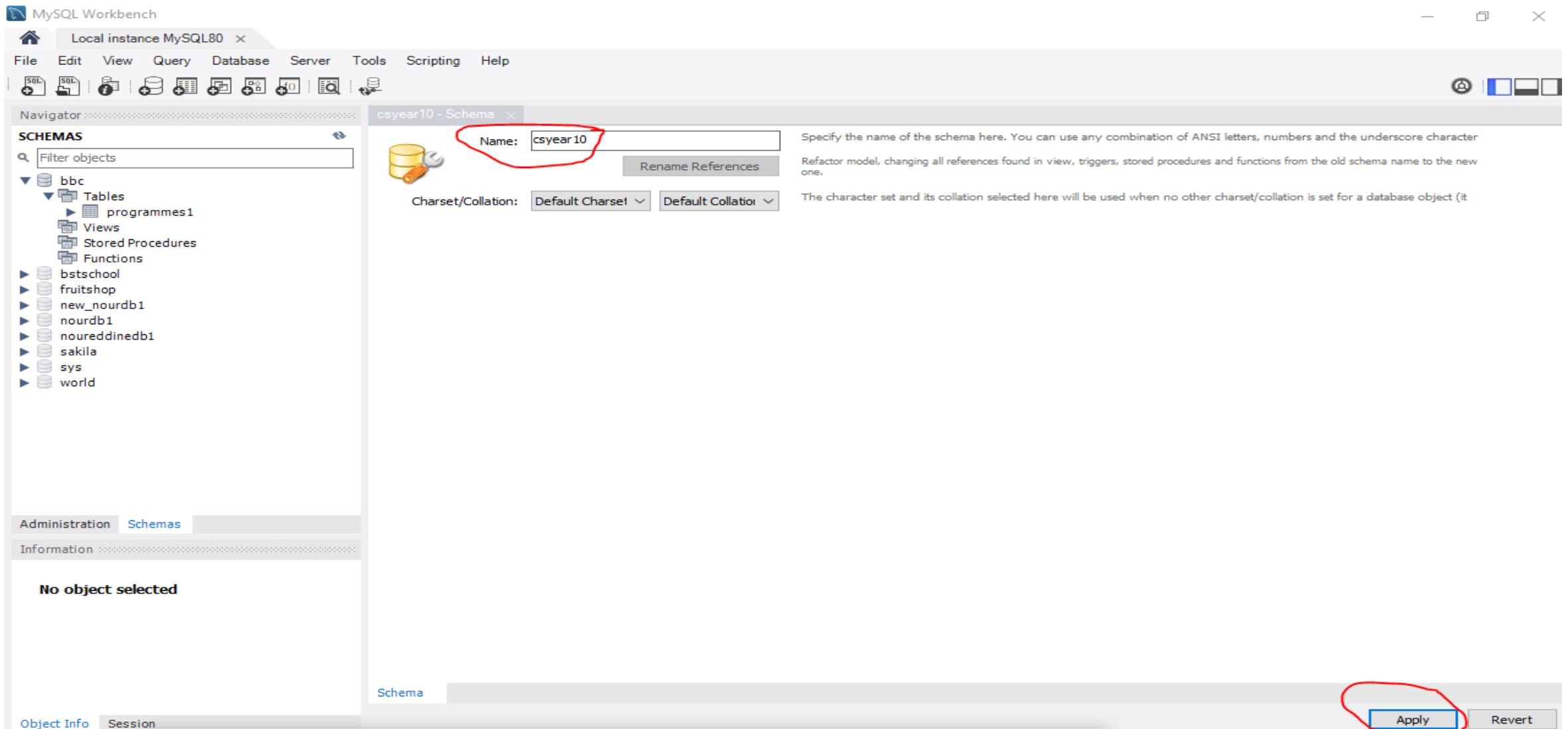
Using MySQL Workbench: Create a new schema it mean a new database

Or right click with your mouse MySQL Workbench and click Create Schema



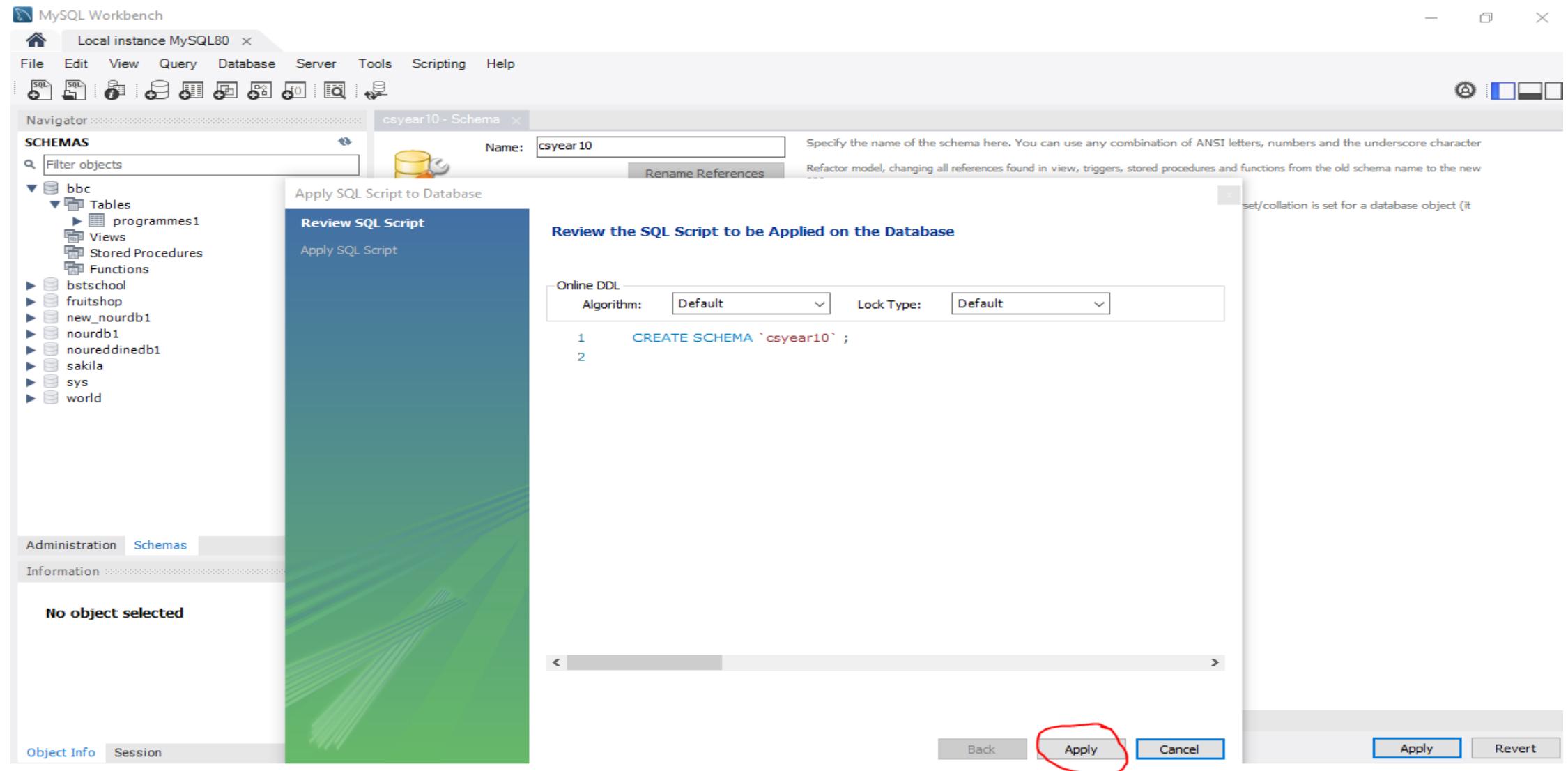
Using MySQL Workbench: Create a new schema it mean a new database

In the Schema name enter csyear10 and click Apply



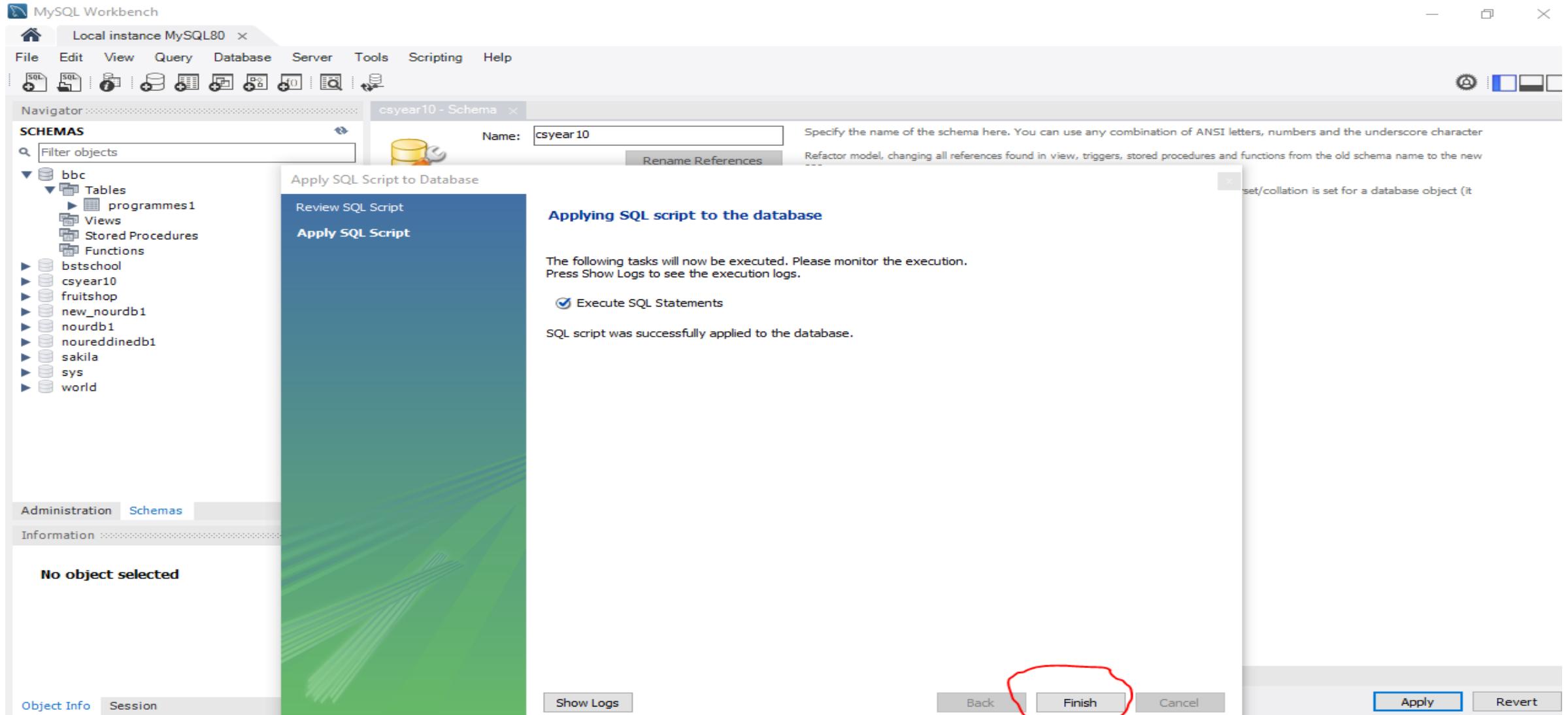
Using MySQL Workbench: Create a new schema it mean a new database

Now the Review is ready to create a schema. Now click Apply as shown below



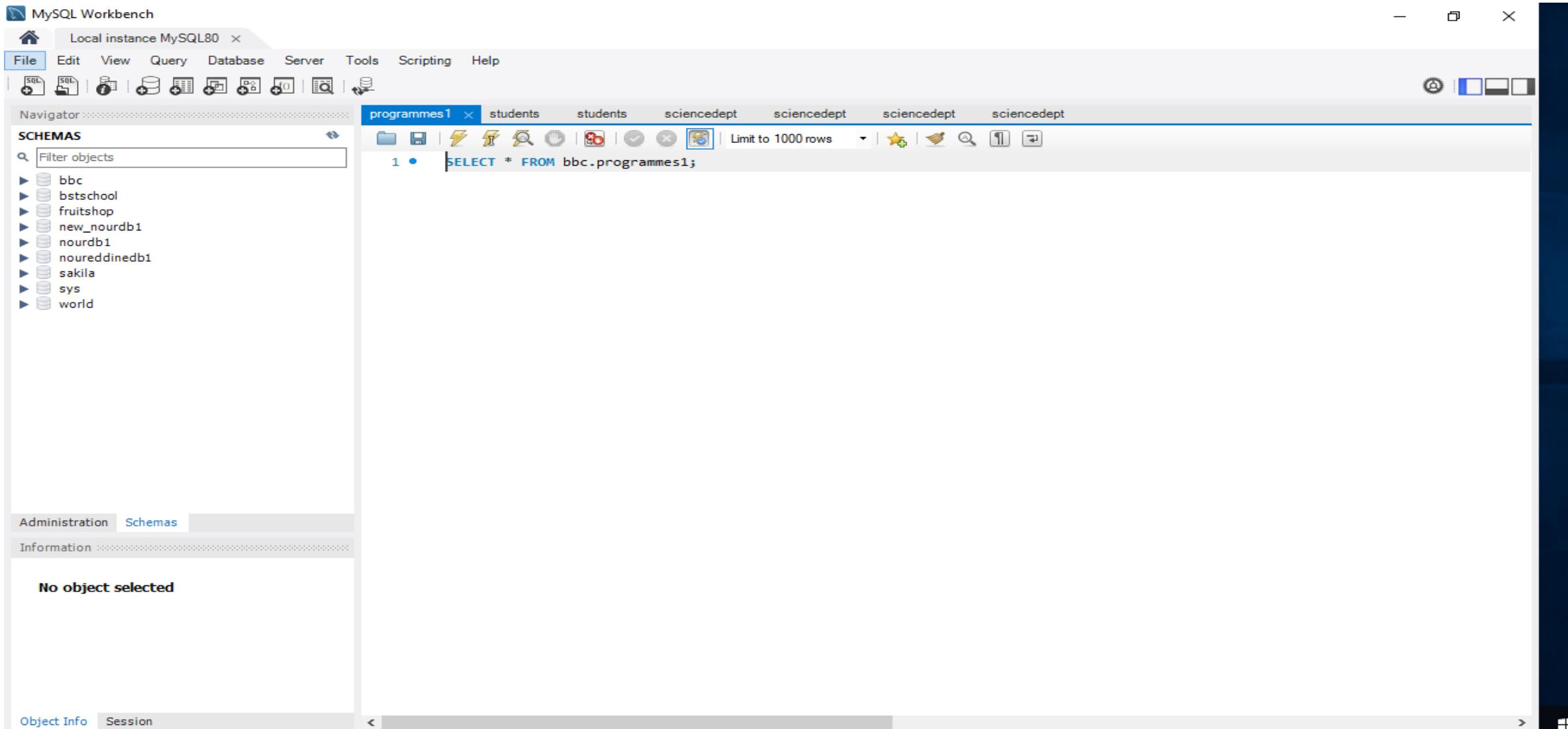
Using MySQL Workbench: Create a new schema it mean a new database

Now click Finish



Using MySQL Workbench: Getting starter

Click Local instance MySQL 8.0



Step by Step practical MySQL

Create a new Table called **Employee**

The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays the schema 'nourdb1' with its tables, views, stored procedures, and functions. The main area shows the 'Employee - Table' configuration for a new table named 'Employee' in the 'nourdb1' schema. The table has four columns: 'EmpID' (INT, PK, NN), 'EmpName' (VARCHAR(60)), 'EmpAge' (INT), and 'EmpDept' (VARCHAR(45)). The 'EmpDept' column is currently selected. Below the table definition, there is a detailed configuration panel for the 'EmpDept' column, including fields for Data Type (VARCHAR(45)), Default value, and Storage options (Virtual, Primary Key, Not Null, Unique, Binary, Unsigned, Zero Fill, Auto Increment, Generated). At the bottom of this panel, the 'Apply' button is circled in red. To the right, the 'Apply SQL Script to Database' dialog is open, showing the generated SQL script:

```
CREATE TABLE `amitdb1`.`employee` (
  `EmpID` INT NOT NULL,
  `EmpName` VARCHAR(60) NOT NULL,
  `EmpAge` INT NOT NULL,
  `EmpDept` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`EmpID`)
);
```

Click Apply Again

Click Finish

Step by Step practical MySQL

Click on the Employee mark to see the visibility of table you created

The screenshot shows the MySQL Workbench interface. The top navigation bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, Help, and a toolbar with various icons. The left sidebar has a 'Navigator' section with 'SCHEMAS' and 'Tables'. Under 'nourdb1', the 'Tables' section is expanded, showing 'employee' (marked with a blue dot), 'Views', 'Stored Procedures', and 'Functions'. The 'employee' table is selected, indicated by a blue border around its row in the tree view. The main query editor window titled 'Query 1' contains the SQL command: 'SELECT * FROM nourdb1.employee;'. Below the query editor is a 'Result Grid' showing a single row of data with columns: EmpID, EmpName, EmpAge, and Empdept, all containing NULL values.

	EmpID	EmpName	EmpAge	Empdept
*	NULL	NULL	NULL	NULL

Step by Step practical MySQL

Now let's **insert** some value record on the Employee table and when you finish click **apply**

The screenshot shows the MySQL Workbench interface. In the center, the 'employee' table is displayed in a grid format. The columns are labeled EmpID, EmpName, EmpAge, and EmpDept. The data rows are:

EmpID	EmpName	EmpAge	EmpDept
1	Swaraj	16	CS
2	Lujain	15	CS
3	Dhyey	16	Math
4	Jeremy	15	Biology
5	Jeremy	17	French
6	Mridula	17	Physics
*	NULL	NULL	NULL

At the bottom right of the interface, there is a toolbar with several icons. The 'Apply' button is highlighted with a red circle.

Step by Step practical MySQL

As you can see now you **inserted** values on your record table and when you finish click **apply** then click **finish**

Apply SQL Script to Database



MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Schemas: fruitshop, new_nourdb1, nourdb1 (Tables: employee, Views, Stored Procedures, Functions), sakila, sys, world

Query 1: employee - Table employee

1 • SELECT * FROM nourdb1.employee;

Result Grid:

EmpID	EmpName	EmpAge	Empdept
1	Swaraj	16	CS
2	Lujain	15	CS
3	Dhyey	16	Math
4	Jeremy	15	Biology
5	Jeremy	17	French
6	Mridula	17	Physics
*	NULL	NULL	NULL

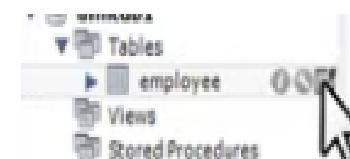
Step by Step practical MySQL

Click here to **refresh** your table

The screenshot shows the MySQL Workbench interface. The title bar says "MySQL Workbench" and "Local instance MySQL80". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The toolbar has various icons for database management. The Navigator pane on the left shows "SCHEMAS" with "nourdb1" expanded, revealing "Tables", "Views", "Stored Procedures", and "Functions". Under "Tables", "employee" is selected. The main area shows a query editor with the SQL command "SELECT * FROM nourdb1.employee;" and a result grid below it. The result grid displays the following data:

	EmpID	EmpName	EmpAge	Empdept
1	Swaraj	16	CS	
2	Lujain	15	CS	
3	Dhyey	16	Math	
4	Jeremy	15	Biology	
5	Jeremy	17	French	
6	Mridula	17	Physics	
*	HULL	HULL	HULL	

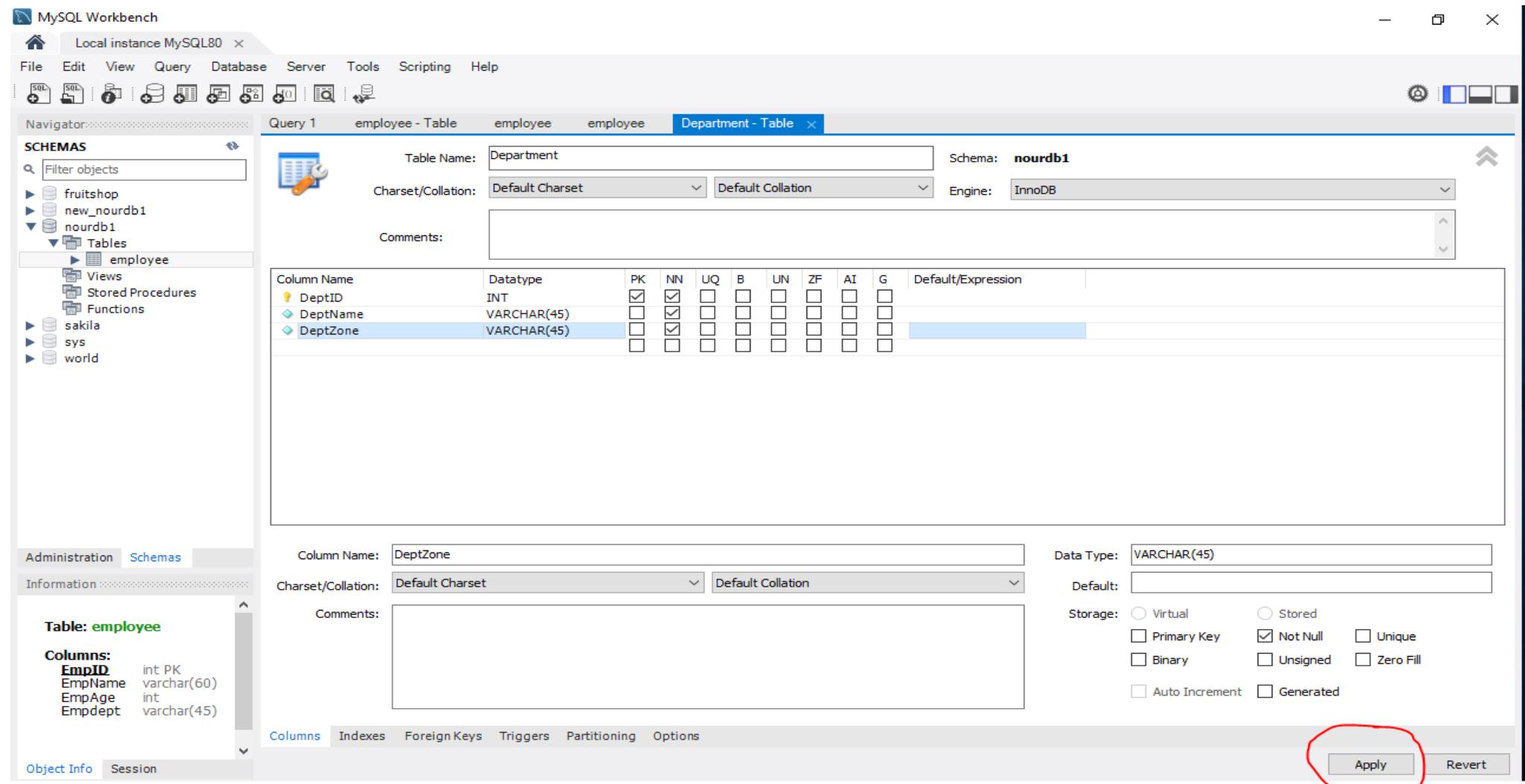
Click on the mark next to employee



Step by Step practical MySQL

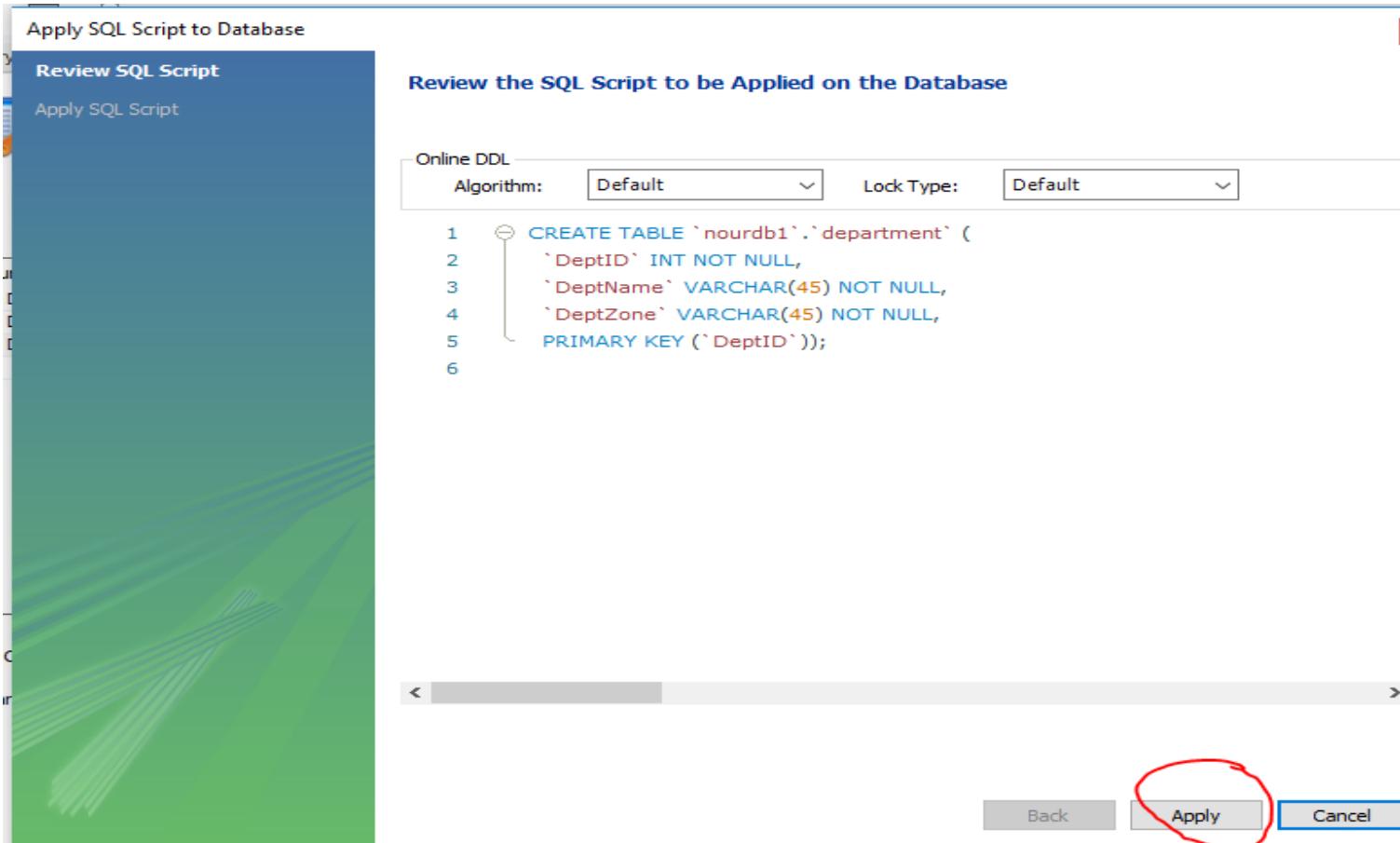
Let's add another table

First Right click on the Employee then click **create a table** and when you finish click **Apply**



Step by Step practical MySQL

Click **Apply** and then click **Finish**



Step by Step practical MySQL

Now you have created two table **Employee** and **Department** as shown below

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The main area has tabs for Navigator, Query 1, employee - Table, employee, employee, and department - Table (which is currently selected). On the left, the Navigator pane shows SCHEMAS with 'fruitshop', 'new_nourdb1', and 'nourdb1'. Under 'nourdb1', there are Tables, Views, Stored Procedures, and Functions. A red circle highlights the 'Tables' section, which contains 'department' and 'employee'. The central workspace shows the 'department - Table' configuration. The 'Table Name' is 'department', 'Schema' is 'nourdb1', 'Charset/Collation' is 'utf8mb4' with 'utf8mb4_0900_ai_ci' selected, and 'Engine' is 'InnoDB'. The 'Comments:' field is empty. Below this, a table defines the columns:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
DeptID	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						
DeptName	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						
DeptZone	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						

Step by Step practical MySQL

Now let's **insert** some value record on the Department table and when you finish click **apply**

The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays the database schema, including the 'nourdb1' database which contains the 'department' table. The main area shows a query editor with the following SQL command:

```
1 •  SELECT * FROM nourdb1.department;
```

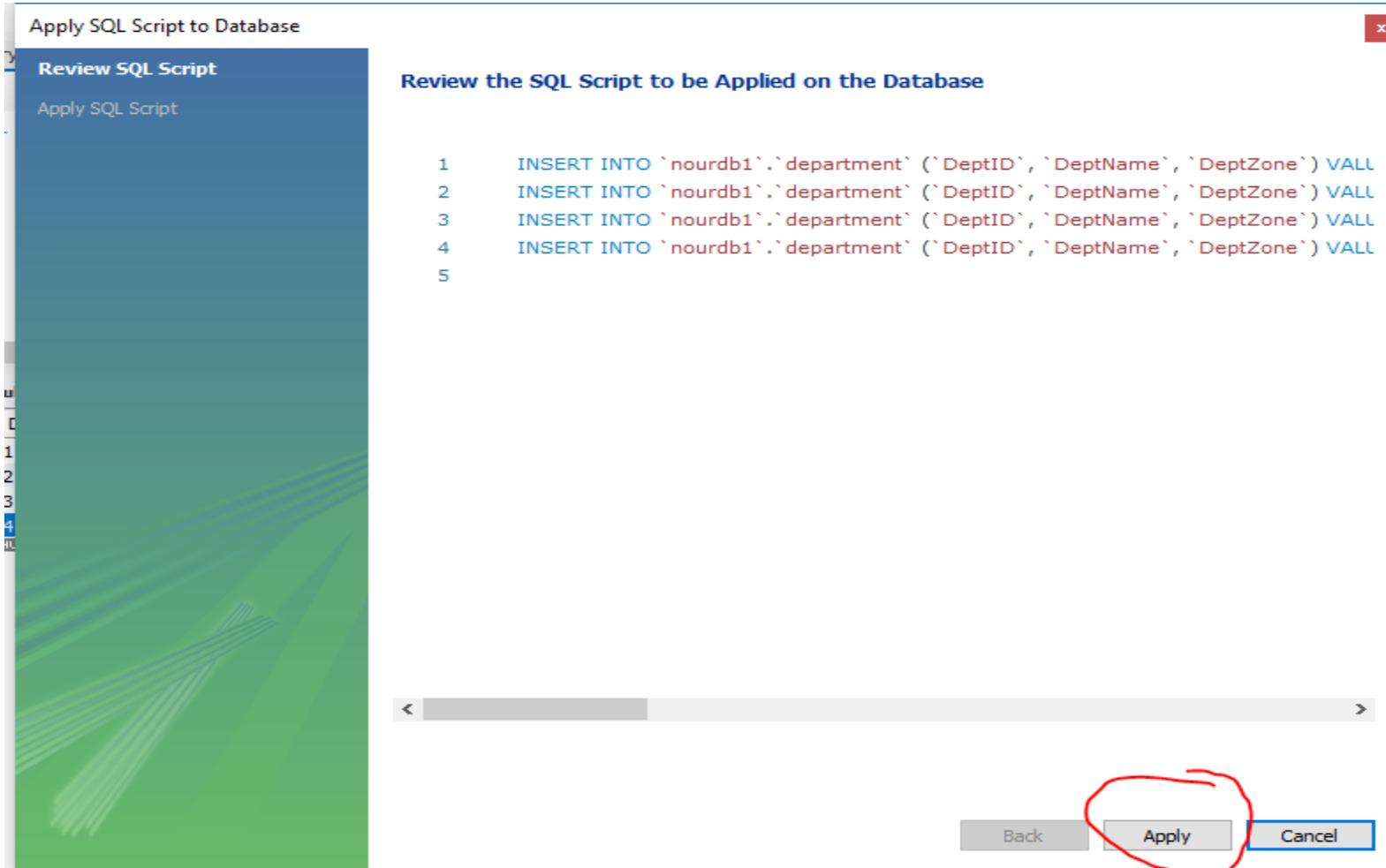
Below the query editor is a Result Grid displaying the current data in the 'department' table:

DeptID	DeptName	DeptZone
1	CS	Kalandar
2	KG	Sayram
3	Science	Kalandar
4	Nord Anglia HQ	London
*	NULL	NULL

A red circle highlights the 'Apply' button at the bottom right of the result grid toolbar. The bottom left of the interface shows the 'Object Info' tab.

Step by Step practical MySQL

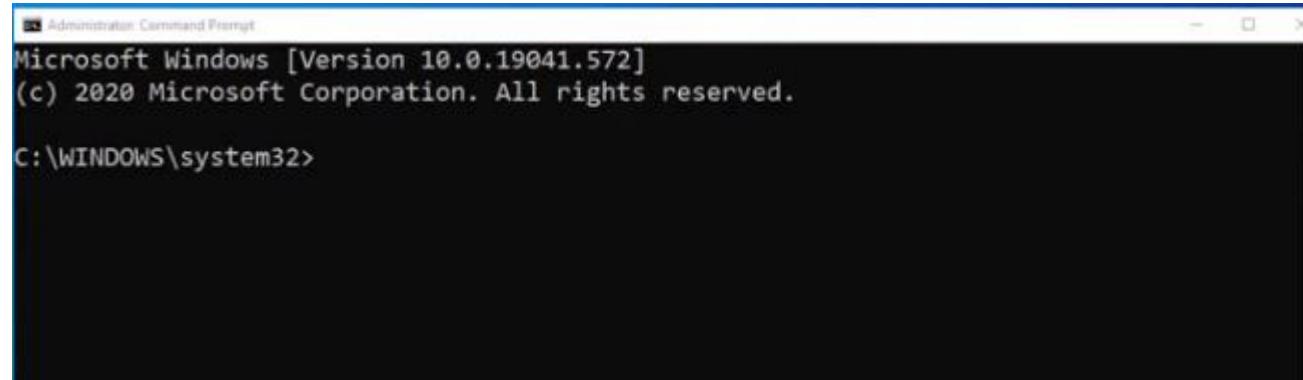
Now click **apply** then click **Finish**



Step by Step practical MySQL

Now Let's start our first SQL programming using **Command prompt**:

1. **Create your first Database with Command prompt**
 - **On search click Command prompt**
 - **Right click on Command prompt and click Run as Administrator**

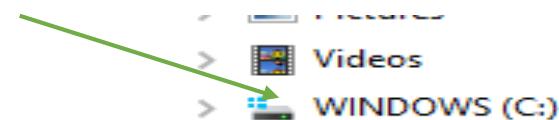


Step by Step practical MySQL

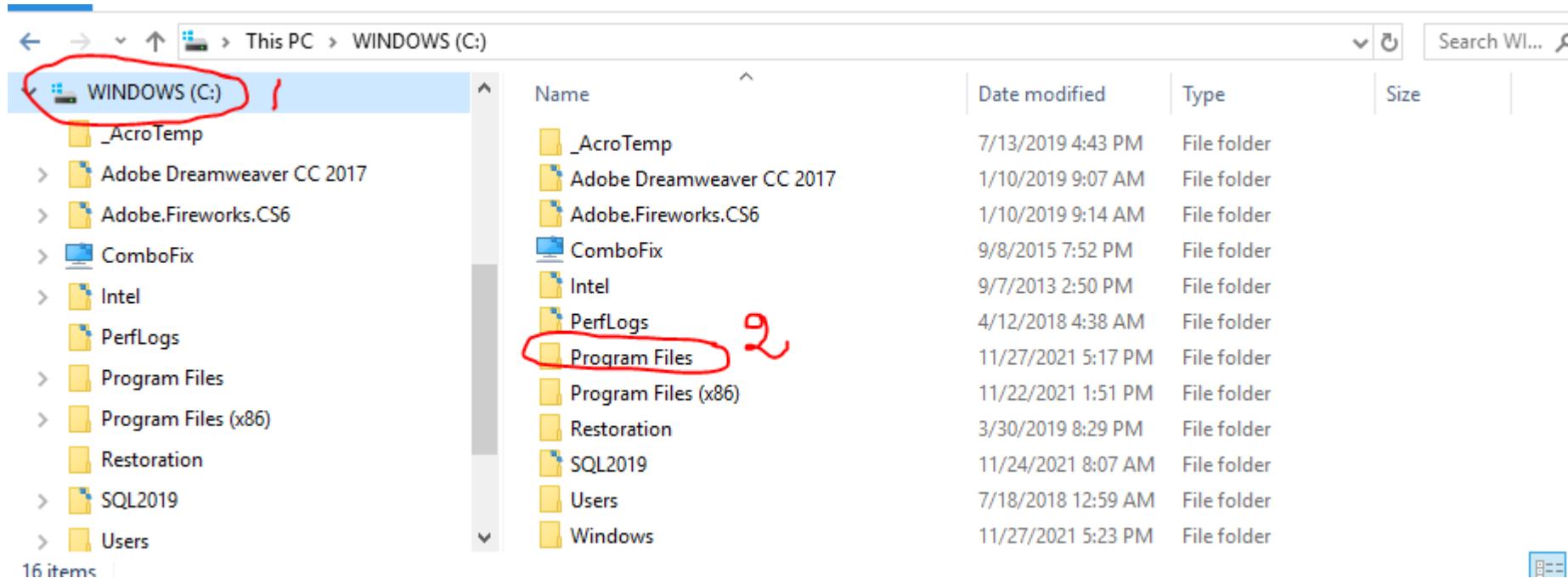
Now Let's start our first SQL programming using **Command prompt**:

2. Create your first Database with Command prompt

- On search click **Command prompt**
- Right click on **Command prompt** and click **Run as Administrator**
- Double click on your C drive as show on the picture:

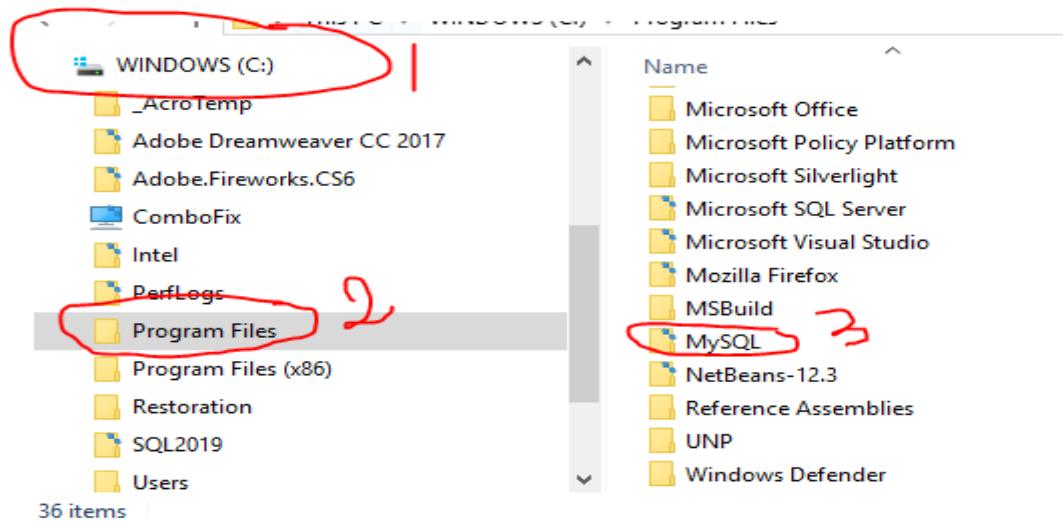


- Double click on **Program Files** as shown below:

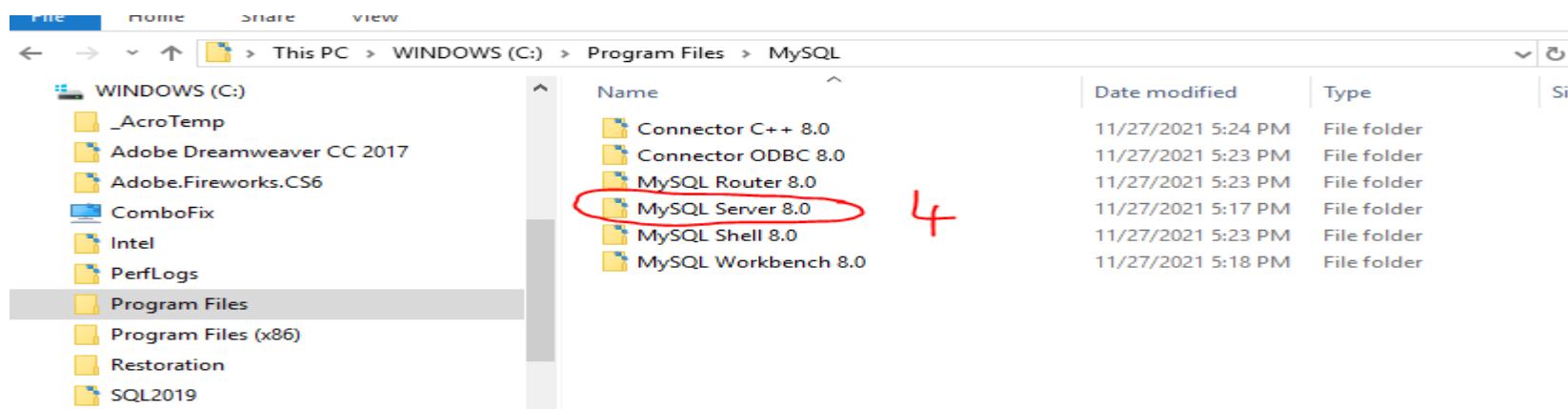


Step by Step practical MySQL

- Scroll down and Find MySQL folder and double click on it

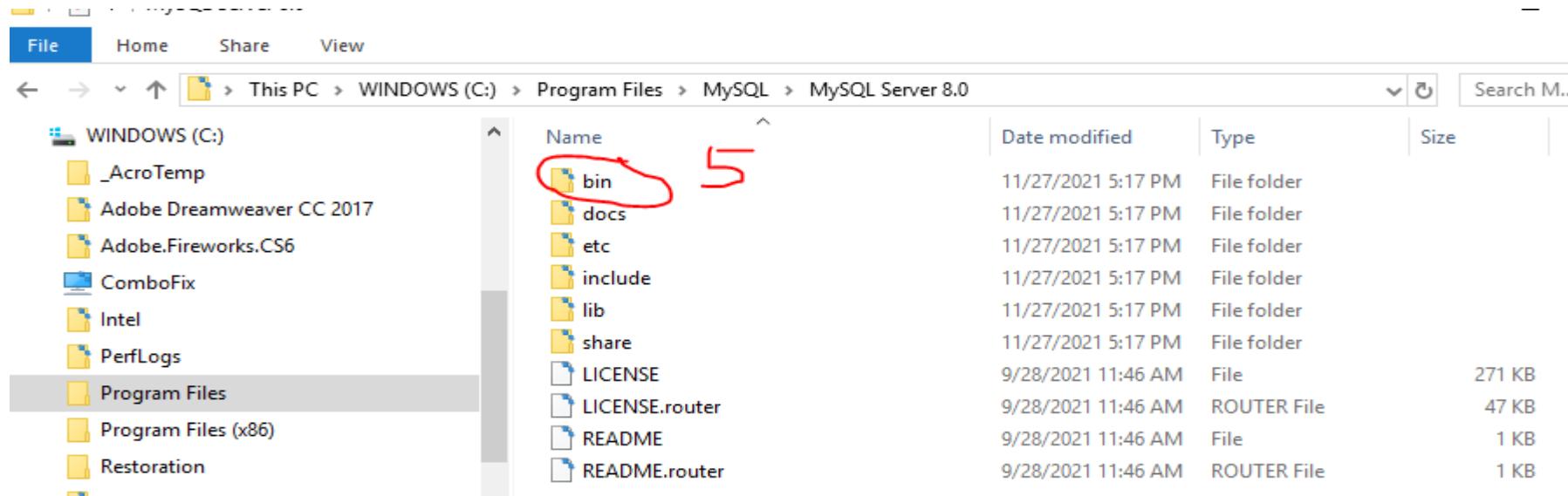


Double click on MySQL Server 8.0



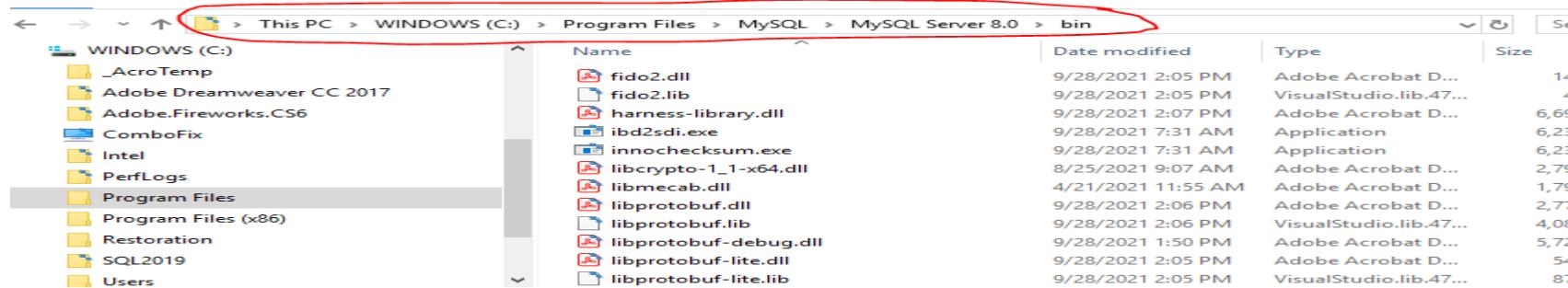
Step by Step practical MySQL

- Double click on bin folder



Now copy the link below on the top of the search bar and past it on command prompt :

C:\Program Files\MySQL\MySQL Server 8.0\bin



Step by Step practical MySQL

Now copy and paste the link below on the top of the search bar on command prompt :

C:\Program Files\MySQL\MySQL Server 8.0\bin

In Command prompt type : cd C:\Program Files\MySQL\MySQL Server 8.0\bin

```
C:\WINDOWS\system32>cd C:\Program Files\MySQL\MySQL Server 8.0\bin
```

In command prompt type: mysql -u root -p and then click enter

Enter the password you created when you download MySQL server and then click enter

```
C:\Program Files\MySQL\MySQL Server 8.0\bin>mysql -u root -p
Enter password: *****
```

Now type: show databases;

```
mysql> show databases;
+-----+
| Database |
+-----+
| fruitshop |
| information_schema |
| mysql |
| new_nourdb1 |
| nourdb1 |
| performance_schema |
| sakila |
| sys |
| world |
+-----+
9 rows in set (0.00 sec)
```

Step by Step practical MySQL

Create your first database:

1- Type:

```
mysql> create database noureddinedb1;
```

2- now you created your first database

```
mysql> create database noureddinedb1;
Query OK, 1 row affected (0.28 sec)
```

3- Type: **show databases;**

```
mysql> show databases;
+-----+
| Database |
+-----+
| fruitshop
| information_schema
| mysql
| new_nourdb1
| nourdb1
| noureddinedb1
| performance_schema
| sakila
| sys
| world
+-----+
10 rows in set (0.10 sec)
```

4- Type: **use noureddinedb1;**

```
mysql> use noureddinedb1;
```

```
mysql> use noureddinedb1;
Database changed
```

Step by Step practical MySQL

Create your first Table:

1- Type:

```
mysql> CREATE TABLE STUDENTS (
    -> StdID int,
    -> FirstName varchar(255),
    -> LastName varchar(255),
    -> StdAge int,
    -> StdSubject varchar(255)
    -> );
Query OK, 0 rows affected (2.55 sec)
```

2- Type: DESC STUDENTS;

```
mysql> DESC STUDENTS;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| StdID | int    | YES  |     | NULL    |       |
| FirstName | varchar(255) | YES  |     | NULL    |       |
| LastName | varchar(255) | YES  |     | NULL    |       |
| StdAge | int    | YES  |     | NULL    |       |
| StdSubject | varchar(255) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+
5 rows in set (0.07 sec)
```

3- INSERT data to your Table Type same as below :

```
mysql> INSERT INTO STUDENTS(StdID, FirstName, LastName, StdAge, StdSubject) VALUES(1, 'SWARAJ', 'Patel', 16, 'AS/A Level Computer Science');
Query OK, 1 row affected (0.18 sec)
```

Step by Step practical MySQL

Create your first Table:

1- Let's see one record to STUDENTS table Type: **SELECT * FROM STUDENTS;**

```
mysql> SELECT * FROM STUDENTS;
+-----+-----+-----+-----+
| StdID | FirstName | LastName | StdAge | StdSubject
+-----+-----+-----+-----+
|     1 | sWARAJ   | Patel    |     16 | AS/A Level Computer Science |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

2- Now add two more record to STUDENTS TABLE Type: **DESC STUDENTS;**

```
mysql> INSERT INTO STUDENTS(StdID, FirstName, LastName, StdAge, StdSubject) VALUES(2,'Lujain','Alotaibi',15,'AS/A Level Computer Science');
Query OK, 1 row affected (0.25 sec)

mysql>
mysql> INSERT INTO STUDENTS(StdID, FirstName, LastName, StdAge, StdSubject) VALUES(3,'Dhyey','Vanraj',16,'AS/A Level Biology');
Query OK, 1 row affected (0.22 sec)

mysql> SELECT *FROM STUDENTS;
+-----+-----+-----+-----+
| StdID | FirstName | LastName | StdAge | StdSubject
+-----+-----+-----+-----+
|     1 | sWARAJ   | Patel    |     16 | AS/A Level Computer Science |
|     2 | Lujain    | Alotaibi |     15 | AS/A Level Computer Science |
|     3 | Dhyey     | Vanraj   |     16 | AS/A Level Biology           |
+-----+-----+-----+-----+
3 rows in set (0.05 sec)
```

Step by Step practical MySQL

Create your first Table:

2- Now let's see the STUDENTS TABLE Type: **SELECT * FROM STUDENTS;**

```
mysql> SELECT * FROM STUDENTS;
+-----+-----+-----+-----+
| StdID | FirstName | LastName | StdAge | StdSubject
+-----+-----+-----+-----+
|     1 | SWARAJ    | Patel     |      16 | AS/A Level Computer Science |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

3- Now insert more records into STUDENTS TABLE Type: **SELECT * FROM STUDENTS;**

```
mysql> INSERT INTO STUDENTS(StdID, FirstName, LastName, StdAge, StdSubject) VALUES(2,'Lujain','Alotaibi',15,'AS/A Level Computer Science');
Query OK, 1 row affected (0.25 sec)

mysql>
mysql> INSERT INTO STUDENTS(StdID, FirstName, LastName, StdAge, StdSubject) VALUES(3,'Dhyey','Vanraj',16,'AS/A Level Biology');
Query OK, 1 row affected (0.22 sec)
```

Now let's see again all the records you inserted into the STUDENTS TABLE Type: **SELECT * FROM STUDENTS;**

```
mysql> SELECT *FROM STUDENTS;
+-----+-----+-----+-----+
| StdID | FirstName | LastName | StdAge | StdSubject
+-----+-----+-----+-----+
|     1 | SWARAJ    | Patel     |      16 | AS/A Level Computer Science |
|     2 | Lujain     | Alotaibi  |      15 | AS/A Level Computer Science |
|     3 | Dhyey      | Vanraj    |      16 | AS/A Level Biology           |
+-----+-----+-----+-----+
3 rows in set (0.05 sec)
```

Step by Step practical MySQL

4- now let's look at our database Type: **show databases;**

```
mysql> show databases;
```

```
+-----+  
| Database          |  
+-----+  
| fruitshop         |  
| information_schema |  
| mysql             |  
| new_nourdb1       |  
| nourdb1           |  
| noureddinedb1     |  
| performance_schema |  
| sakila            |  
| sys               |  
| world              |  
+-----+  
10 rows in set (10.54 sec)
```

5 – Make sure you type the following SQL QUERIES BELOW:

```
mysql> SELECT * FROM STUDENTS;  
+-----+  
| StdID | FirstName | LastName | StdAge | StdSubject |  
+-----+  
| 1    | sWARAJ   | Patel    | 16    | AS/A Level Computer Science |  
| 2    | Lujain    | Alotaibi | 15    | AS/A Level Computer Science |  
| 3    | Dhyey     | Vanraj   | 16    | AS/A Level Biology        |  
+-----+  
3 rows in set (0.00 sec)  
  
mysql> SELECT * FROM STUDENTS WHERE LastName = 'Patel' AND StdAge='16';  
+-----+  
| StdID | FirstName | LastName | StdAge | StdSubject |  
+-----+  
| 1    | sWARAJ   | Patel    | 16    | AS/A Level Computer Science |  
+-----+  
1 row in set (0.00 sec)
```

Step by Step practical MySQL Activity 2

1- Now create a new database called BST

2- Create three table Customer, Product and Orders

3 – On the first table called Customer enter the value below:

Customer table:

The customer table gives customers a unique Customer ID (the primary key for this table) and shows customer details, ie name, address and phone number:

Customer ID	First name	Surname	Address	Phone number
02942	Rebecca	Johnson	49 Drew Road	029 381834
02943	Mushtaq	Aqbar	28 Lyttleton Lane	028 282738

Product table:

The product table gives details about the products. The Product ID is the primary key for this table

Product ID	Product type	Colour	Size	Cost
284758	Jeans	Blue	28	£14.99
384957	Shoes	Brown	6	£12.99
483927	Jumper	Red	M	£29.99
489320	Shirt	Blue	M	£33.99
839258	Socks	White	6	£10.00

Step by Step practical MySQL Activity2

Orders table:

In the orders table, each order has a unique Order number (the primary key for this table). The table also includes customer ID (the primary key of the customer table) and product ID (the primary key of the product table) as **foreign keys**, but does not need to include all details about customers and products as these are stored in the Customer and Product tables.

Order number	Customer ID	Product ID	Quantity	Total cost
59876	02942	284758	2	£29.98
59877	02942	384957	3	£38.97
59878	02942	483927	1	£29.99
59879	02943	489320	3	£101.97
59880	02943	839258	2	£20.00

Practical working with SQL: Step 1: Activity 3

Create your first database: called BBC

Type: show databases;

```
mysql> show databases;
+-----+
| Database
+-----+
| bbc
| fruitshop
| information_schema
| mysql
| new_nourdb1
| nourdb1
| noureddinedb1
| performance_schema
| sakila
| sys
| world
+-----+
11 rows in set (0.00 sec)
```

```
mysql> CREATE DATABASE BBC;
Query OK, 1 row affected (0.51 sec)
```

- Type: use BBC;

```
mysql> use BBC;
Database changed.
```

- Creating a table:

```
CREATE TABLE Programmes
(ID int(2),
Title varchar(20),
Genre varchar(20),
Duration int(3));
```

Type: DESC STUDENTS;

```
mysql> DESC bbc.programmes1;
+-----+-----+-----+-----+-----+
| Field | Type  | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| ID   | int   | NO  | PRI | NULL    |       |
| Title | varchar(20) | NO  |     | NULL    |       |
| Genre | varchar(20) | NO  |     | NULL    |       |
| Duration | varchar(45) | NO  |     | NULL    |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

Practical working with SQL: Step 2: Activity 3

- **INSERT data to your Table and Type same as below:**

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90

ID	Title	Gender	Duration
1	Eastender	Drama	30
2	Newsnight	Currenrt affaire	50
3	Voices	Enterainmentt	75
4	Blue Peter	Childrens	25
5	Wild Brazil	Nature	60
6	Sherlock	Drama	90

- Now let's do some MySQL Queries on the **bbc programmes1** TABLE Type: **SELECT ID, Gender * FROM bbc.programmes1;**

```
mysql> SELECT ID, Gender FROM bbc.programmes1;
+----+-----+
| ID | Gender |
+----+-----+
| 1  | Drama   |
| 2  | Currenrt affaire |
| 3  | Enterainmentt |
| 4  | Childrens |
| 5  | Nature   |
| 6  | Drama   |
+----+-----+
6 rows in set (0.00 sec)
```

Practical working with SQL: Step 3: Activity 3

The **SQl WHERE** statement is used to isolate one **record** or several records with similar **attributes**.

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90

The following code searches the Title **field** of the table to find the words 'The Voice'

```
SELECT Programmes.ID, Programmes.Title, Programmes.Genre,  
Programmes.Duration  
FROM Programmes  
WHERE ((Programmes.Title)="The Voice");
```

- the **WHERE** statement specifies which text to look for

This table shows the results from this query:

ID	Title	Genre	Duration
03	The Voice	Entertainment	75

The SQL SELECT Statement (SELECT...FROM... statement)

The select statement is used to select data from the database. Type SQL SELECT statement below:

```
mysql> SELECT *FROM bbc.programmes1;
+----+-----+-----+
| ID | Ttile      | Gender          | Duration |
+----+-----+-----+
| 1  | Eastender   | Drama           | 30
| 2  | Newsnight   | Currenrt affaire | 50
| 3  | Voices       | Enterainmentt   | 75
| 4  | Blue Peter   | Childrens        | 25
| 5  | Wild Brazil  | Nature           | 60
| 6  | Sherlock     | Drama           | 90
+----+-----+-----+
6 rows in set (0.00 sec)
```

```
mysql> SELECT ID, Gender FROM bbc.programmes1;
+----+-----+
| ID | Gender      |
+----+-----+
| 1  | Drama        |
| 2  | Currenrt affaire |
| 3  | Enterainmentt |
| 4  | Childrens     |
| 5  | Nature         |
| 6  | Drama          |
+----+-----+
6 rows in set (0.00 sec)
```

The SQL SELECT Statement (SELECT...FROM... WHERE... clause)

To go a step further, the WHERE clause can be used. The WHERE clause will only return records that meet a specific condition: Type SQL SELECT statement below:

```
mysql> SELECT *FROM bbc.programmes1 WHERE Gender='Drama';
+----+-----+-----+
| ID | Ttile | Gender | Duration |
+----+-----+-----+
| 6  | Sherlock | Drama | 98      |
+----+-----+-----+
1 row in set (0.00 sec)
```

```
* FROM BBC.programmes1 WHERE ID = 4; at line
mysql> SELECT *FROM bbc.programmes1
-> WHERE ID = 4;
+----+-----+-----+-----+
| ID | Ttile       | Gender     | Duration |
+----+-----+-----+-----+
| 4  | Blue Peter | Childrens | 25        |
+----+-----+-----+-----+
1 row in set (0.00 sec)
```

Using AND, OR, <, > and = operators.

It is possible to use the AND, OR <, > and = operators with a WHERE clause. This will refine a search further.

AND

Using AND allows the database developer to specify more than one condition that must be met. In this example, two conditions are specified.

```
mysql> SELECT ID, Ttile, Gender, Duration FROM bbc.programmes1
      -> WHERE Duration = 25 AND Ttile = 'Blue Peter';
+----+-----+-----+-----+
| ID | Ttile       | Gender     | Duration |
+----+-----+-----+-----+
|  4 | Blue Peter | Childrens | 25        |
+----+-----+-----+-----+
1 row in set (0.04 sec)
```

OR

Using OR allows the database developer to specify a series of conditions, where only one of the conditions needs to be met.

```
mysql> SELECT ID, Ttile, Gender, Duration FROM bbc.programmes1
      -> WHERE Ttile = 'Eastender' OR Duration >30;
+----+-----+-----+-----+
| ID | Ttile       | Gender     | Duration |
+----+-----+-----+-----+
|  1 | Eastender   | Drama      | 30        |
|  2 | Newsnight   | Current affaire | 50        |
|  3 | Voices      | Entertainment | 75        |
|  5 | Wild Brazil | Nature     | 60        |
|  6 | Sherlock    | Drama      | 90        |
+----+-----+-----+-----+
5 rows in set (0.04 sec)
```

Using AND, OR, <, > and = operators.

Operators

<	Less than
>	Greater than
=	Equals

Make sure you type the below SQL statement:

```
mysql> SELECT ID,Ttitle FROM bbc.programmes1
-> WHERE ID =2 OR Duration <60;
+-----+
| ID | Ttitle |
+-----+
| 1 | Eastender |
| 2 | Newsnight |
| 4 | Blue Peter |
+-----+
3 rows in set (0.00 sec)
```

```
mysql> SELECT ID,Ttitle FROM bbc.programmes1
-> WHERE ID = 2 OR Duration = 60;
+-----+
| ID | Ttitle |
+-----+
| 2 | Newsnight |
| 5 | Wild Brazil |
+-----+
2 rows in set (0.00 sec)
```

ORDER BY.

ORDER BY is an SQL Keyword used to allow the ordering of results in either ascending (**ASC**) or descending order (**DESC**) .

```
mysql> SELECT Ttile, Gender FROM bbc.programmes1
-> ORDER BY Gender ASC;
+-----+-----+
| Ttile      | Gender
+-----+-----+
| Blue Peter | Childrens
| Newsnight  | Currenrt affaire
| Sherlock   | Drama
| Eastender   | Drama
| Voices     | Enterainmentt
| Wild Brazil | Nature
+-----+-----+
6 rows in set (0.04 sec)
```

To sort in descending order, add **DESC** at the end of the statement.

```
mysql> SELECT Ttile, Gender FROM bbc.programmes1
-> ORDER BY Gender DESC;
+-----+-----+
| Ttile      | Gender
+-----+-----+
| Wild Brazil | Nature
| Voices     | Enterainmentt
| Eastender   | Drama
| Sherlock   | Drama
| Newsnight  | Currenrt affaire
| Blue Peter | Childrens
+-----+-----+
6 rows in set (0.00 sec)
```

Sorting multiple columns.

It is possible to apply a sort on more than one column.

The first sort would be applied in its entirety, with the second sort only applied if two records in the column used for the first sort have the same value.

```
mysql> SELECT Ttile, Gender, Duration FROM bbc.programmes1
    -> ORDER BY Gender DESC, Duration ASC;
+-----+-----+-----+
| Ttile      | Gender        | Duration |
+-----+-----+-----+
| Wild Brazil | Nature          | 60
| Voices      | Enterainmentt | 75
| Eastender   | Drama           | 30
| Sherlock    | Drama           | 90
| Newsnight   | Currenrt affaire | 50
| Blue Peter  | Childrens       | 25
+-----+-----+-----+
6 rows in set (0.00 sec)
```

INSERT INTO

The INSERT INTO statement is used to add a record to a table.

The original table called 'Pupil' contains the following records:

Type: **use noureddinedb1;**

```
mysql> use noureddinedb1;
```

Type: **DESC STUDENTS;**

```
mysql> DESC STUDENTS;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| StdID | int    | YES  |     | NULL    |       |
| FirstName | varchar(255) | YES  |     | NULL    |       |
| LastName | varchar(255) | YES  |     | NULL    |       |
| StdAge | int    | YES  |     | NULL    |       |
| StdSubject | varchar(255) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

Type: **SELECT *FROM STUDENTS;**

```
mysql> SELECT*FROM STUDENTS;
+-----+-----+-----+-----+
| StdID | FirstName | LastName | StdAge | StdSubject |
+-----+-----+-----+-----+
| 1 | sWARAJ | Patel | 16 | AS/A Level Computer Science |
| 2 | Lujain | Alotaibi | 15 | AS/A Level Computer Science |
| 3 | Dhyey | Vanraj | 16 | AS/A Level Biology |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

StdID	FirstName	LastName	StdAge	StdSubject
1	sWARAJ	Patel	16	AS/A Level Computer Science
2	Lujain	Alotaibi	15	AS/A Level Computer Science
3	Dhyey	Vanraj	16	AS/A Level Biology

INSERT INTO.

To add a new record, the following SQL code could be used:

Type: `INSERT INTO STUDENTS(StdID,FirstName,LastName,StdAge,StdSubject) VALUES(4,'Murat','Daminov',16,'AS/A Level Physics');`

```
mysql> INSERT INTO STUDENTS(StdID,FirstName,LastName,StdAge,StdSubject) VALUES(4,'Murat','Daminov',16,'AS/A Level Physics');
Query OK, 1 row affected (0.38 sec)
```

Type: **SELECT *FROM STUDENTS;**

```
mysql> SELECT*FROM STUDENTS;
+-----+-----+-----+-----+
| StdID | FirstName | LastName | StdAge | StdSubject
+-----+-----+-----+-----+
|     1 | sWARAJ   | Patel    |      16 | AS/A Level Computer Science |
|     2 | Lujain    | Alotaibi |      15 | AS/A Level Computer Science |
|     3 | Dhyey     | Vanraj   |      16 | AS/A Level Biology           |
|     4 | Murat     | Daminov  |      16 | AS/A Level Physics          |
+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

UPDATE

The UPDATE statement will allow a database developer to update the records held in a table.

Usually, the UPDATE statement is used with a WHERE clause.

The WHERE clause identifies the specific records that are to be updated. If the WHERE clause is not used, all values in a column will be updated.

Type: **UPDATE STUDENTS**

SET StdAge =18;

```
mysql> UPDATE STUDENTS  
      -> SET StdAge =18;
```

Now type to see the new update table : **SELECT *FROM STUDENTS;**

```
mysql> SELECT*FROM STUDENTS;  
+-----+-----+-----+-----+-----+  
| StdID | FirstName | LastName | StdAge | StdSubject |  
+-----+-----+-----+-----+-----+  
|    1 | SWARAJ   | Patel     |    18 | AS/A Level Computer Science |  
|    2 | Lujain    | Alotaibi  |    18 | AS/A Level Computer Science |  
|    3 | Dhyey     | Vanraj    |    18 | AS/A Level Biology          |  
|    4 | Murat     | Daminov   |    18 | AS/A Level Physics          |  
+-----+-----+-----+-----+  
4 rows in set (0.00 sec)
```

DELETE

To delete the record for Tom Glacney, the following SQL would be used:

Type to delete a row record :

```
mysql> DELETE FROM STUDENTS  
      -> WHERE StdAge= 18 AND StdSubject = 'AS/A Level Physics';  
Query OK, 1 row affected (0.18 sec)
```

Now type to see the deleted record on the students table : **SELECT *FROM STUDENTS;**

```
mysql> SELECT*FROM STUDENTS;  
+-----+-----+-----+-----+  
| StdID | FirstName | LastName | StdAge | StdSubject          |  
+-----+-----+-----+-----+  
|     1 | sWARAJ   | Patel    |     18 | AS/A Level Computer Science |  
|     2 | Lujain    | Alotaibi |     18 | AS/A Level Computer Science |  
|     3 | Dhyey     | Vanraj   |     18 | AS/A Level Biology        |  
+-----+-----+-----+-----+  
3 rows in set (0.07 sec)
```

Practical working with SQL: Step 3: Activity 3

Alternatively, you could find all of the programmes which are less than 30 minutes long using this code:

```
...
WHERE ((Programmes.Duration)<30);
```

This table shows the results from this query:

ID	Title	Genre	Duration
04	Blue Peter	Children's	25

Wildcards

The **wildcard** uses the ***** symbol, and is used in place of any number of unknown characters.

For example, the following code searches for all programmes with the letter **i** in the title:

```
...
WHERE ((Programmes.Title) LIKE "*i*");
```

This table shows the results from this query:

ID	Title	Genre	Duration
02	Newshight	Current Affairs	50
03	The Voice	Entertainment	75
05	Wild Brazil	Nature	60

Adding and editing data with SQL Activity 3

SQL can also be used to add and edit the data stored on an SQL server.

Adding records

To add new data you use the function **INSERT INTO**.

If a new BBC programme is created and you want to add it to the **database**, then you would need to use the **INSERT INTO** function followed by the **VALUES** separated by a comma:

```
INSERT INTO Programmes  
VALUES (07, "Top Gas", "Entertainment", 60);
```

Note that quotes surround string entries. Numbers do not have quotes.

After inputting this code the table would look like this:

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90
07	Top Gas	Entertainment	60

Adding and editing data with SQL Activity 3

SQL also has the **UPDATE** function for editing data.

In this example, there was an error with the previous entry and you need to change the name from "Top Gas" to "Top Gear". You need to:

1. Identify the table to be updated using **UPDATE - UPDATE Programmes**
2. Identify what the **field** needs to be changed to using **SET - SET Programmes.Title = "Top Gear"**
3. Identify which **record** needs to be updated using **WHERE - WHERE Programmes.ID = 07**

In full, this is:

```
UPDATE Programmes  
SET Programmes.Title = "Top Gear"  
WHERE Programmes.ID = 07;
```

This will go to the **Programmes** table, find the programme with an **ID** of 07 and change the **Title** field to **Top Gear**. The amended table will look like this:

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90
07	Top Gear	Entertainment	60

Adding and editing data with SQL Activity 3 - Update Data in a MySQL Database

SQL also has the **UPDATE** function or The **UPDATE Statement** for editing data.

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90
07	Top Gear	Entertainment	60

Relational Model Concepts in DBMS:

Attribute: Each column in a Table. Attributes are the properties which define a relation. e.g., Student_Rollno, NAME,etc.

Tables – In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

Tuple – It is nothing but a single row of a table, which contains a single record.

Relation Schema: A relation schema represents the name of the relation with its attributes.

Degree: The total number of attributes which in the relation is called the degree of the relation.

Cardinality: Total number of rows present in the Table.

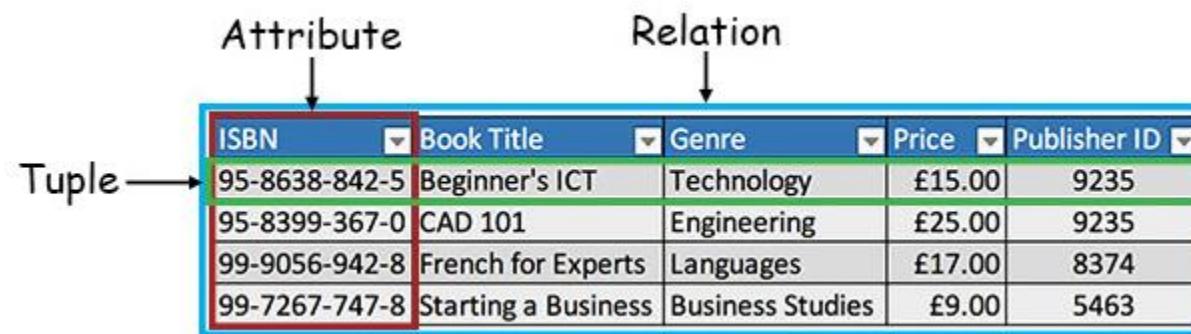
Column: The column represents the set of values for a specific attribute.

Relation instance – Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples.

Relation key – Every row has one, two or multiple attributes, which is called relation key.

Attribute domain – Every attribute has some pre-defined value and scope which is known as attribute domain. Example: The domain of Marital Status has a set of possibilities: Married, Single, Divorced.

Formal Terms (Codd)	Alternative 1	Alternative 2
Relation	Table	File
Tuple	Row	Record
Attribute	Column	Field



Relational Model Concepts in DBMS:

Table also called Relation

© guru99.com		
CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Primary Key

Domain
Ex: NOT NULL

Column OR Attributes

Total # of column is Degree

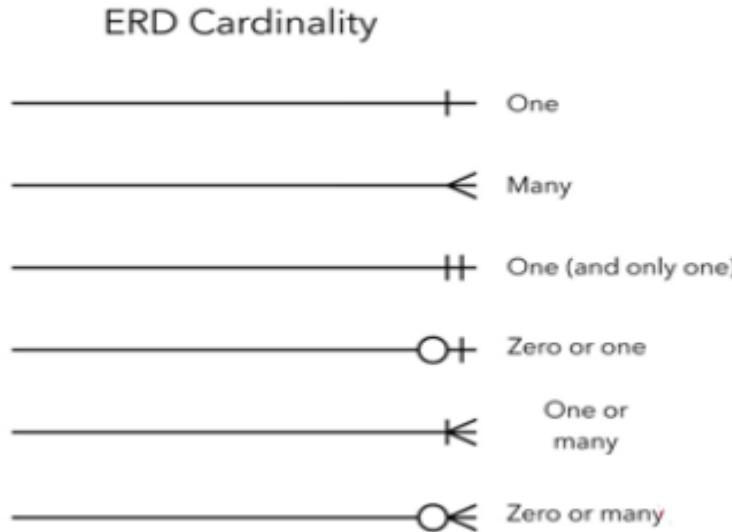
Tuple OR Row

Total # of rows is Cardinality

Entity –Relationship (E-R) Diagram:

Relationship can take several cardinality forms;

- one – to-one, 1:1
- one – to – many : 1:m
- Many-to-one: m:1
- Many-to-many: m:m



Customer	
Key	Customer_ID
Key	FirstName
Key	LastName
Key	Street
Key	City
Key	Zip
Key	Phone

Entities

Order	
PK	Order_number
Key	Customer_ID
Key	Customer_name*
Key	To_street
Key	To_city
Key	To-state
Key	To-zip
Key	Ship_date

Product	
Key	Product_ID
Key	Quantity
Key	Product_type

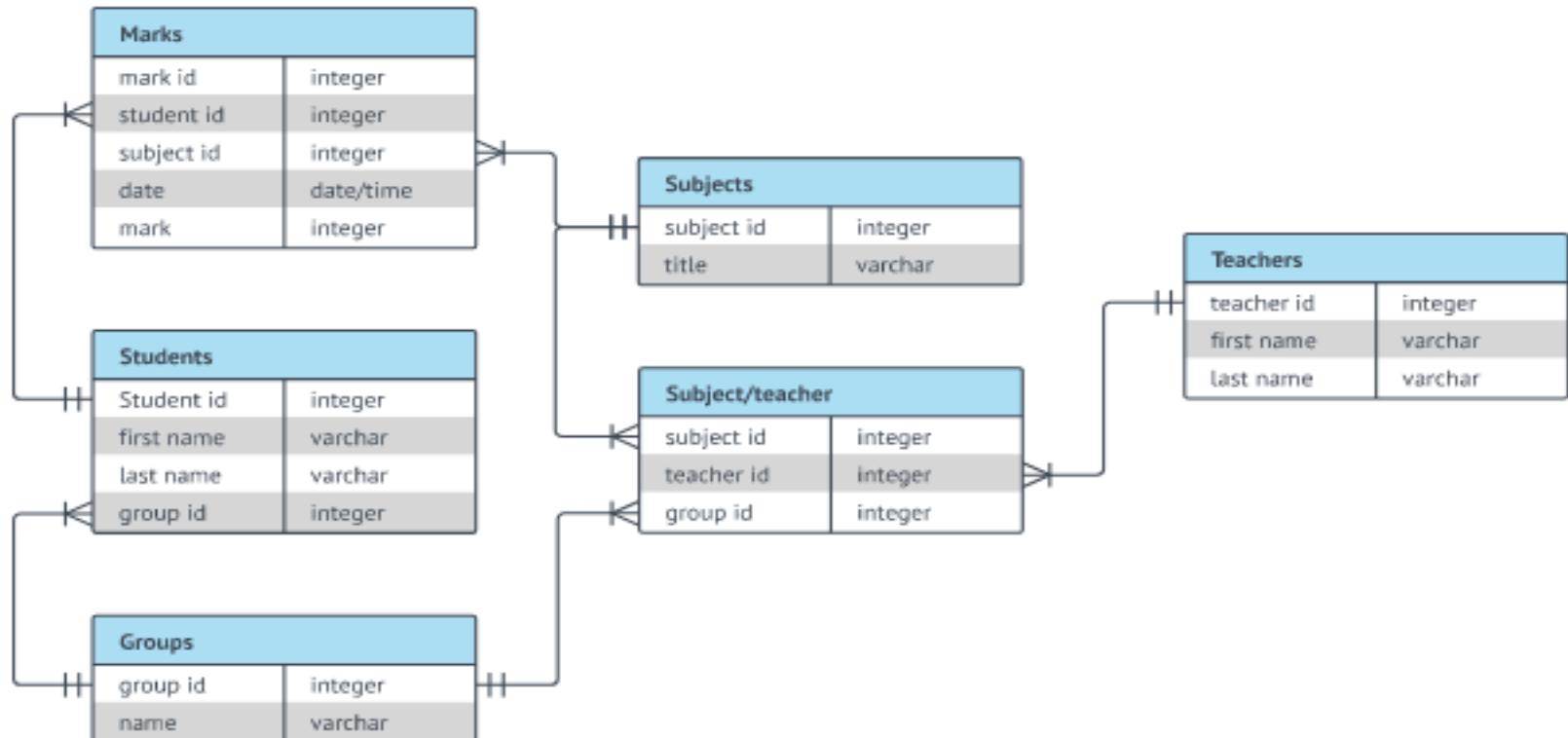
Attributes

Entity –Relationship (E-R) Diagram:

What is an ER diagram?

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

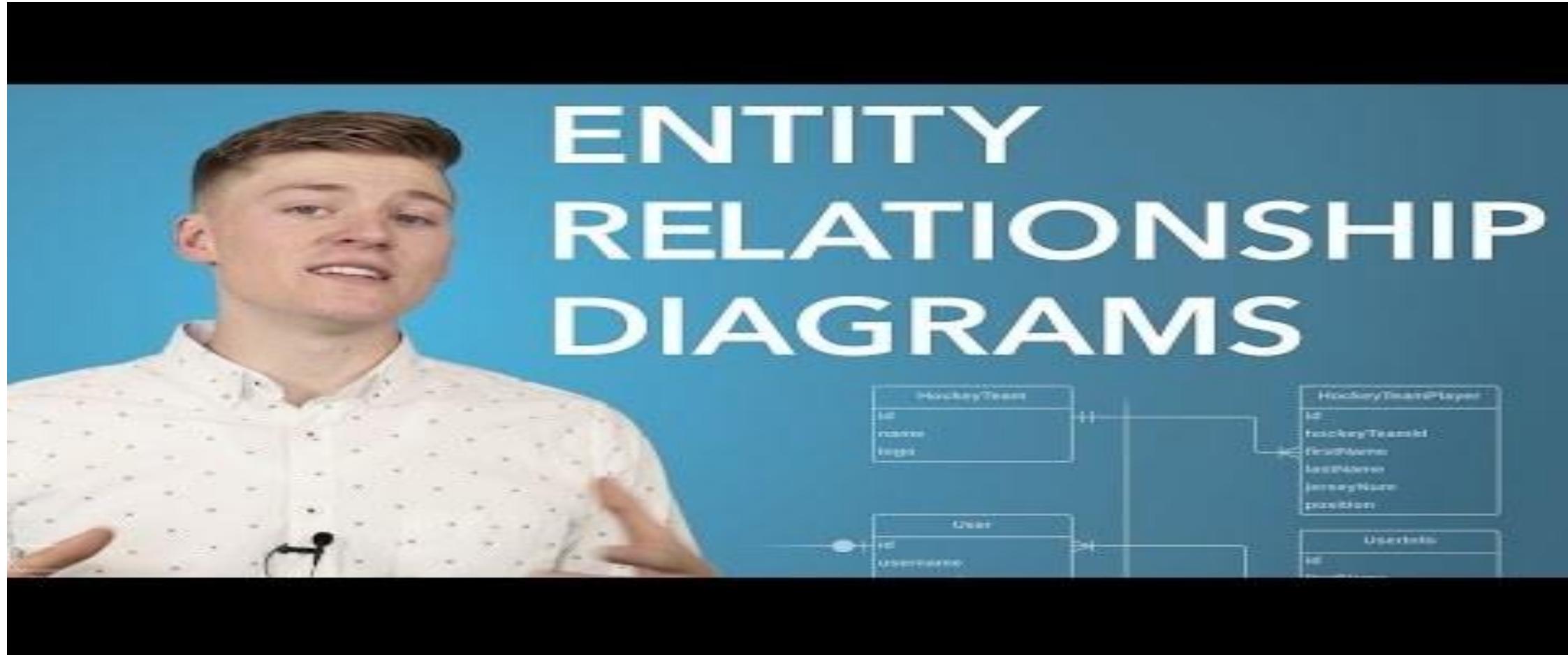
ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.



Entity Relationship Diagrams.

Watch the video for the Entity Relationship Diagrams:

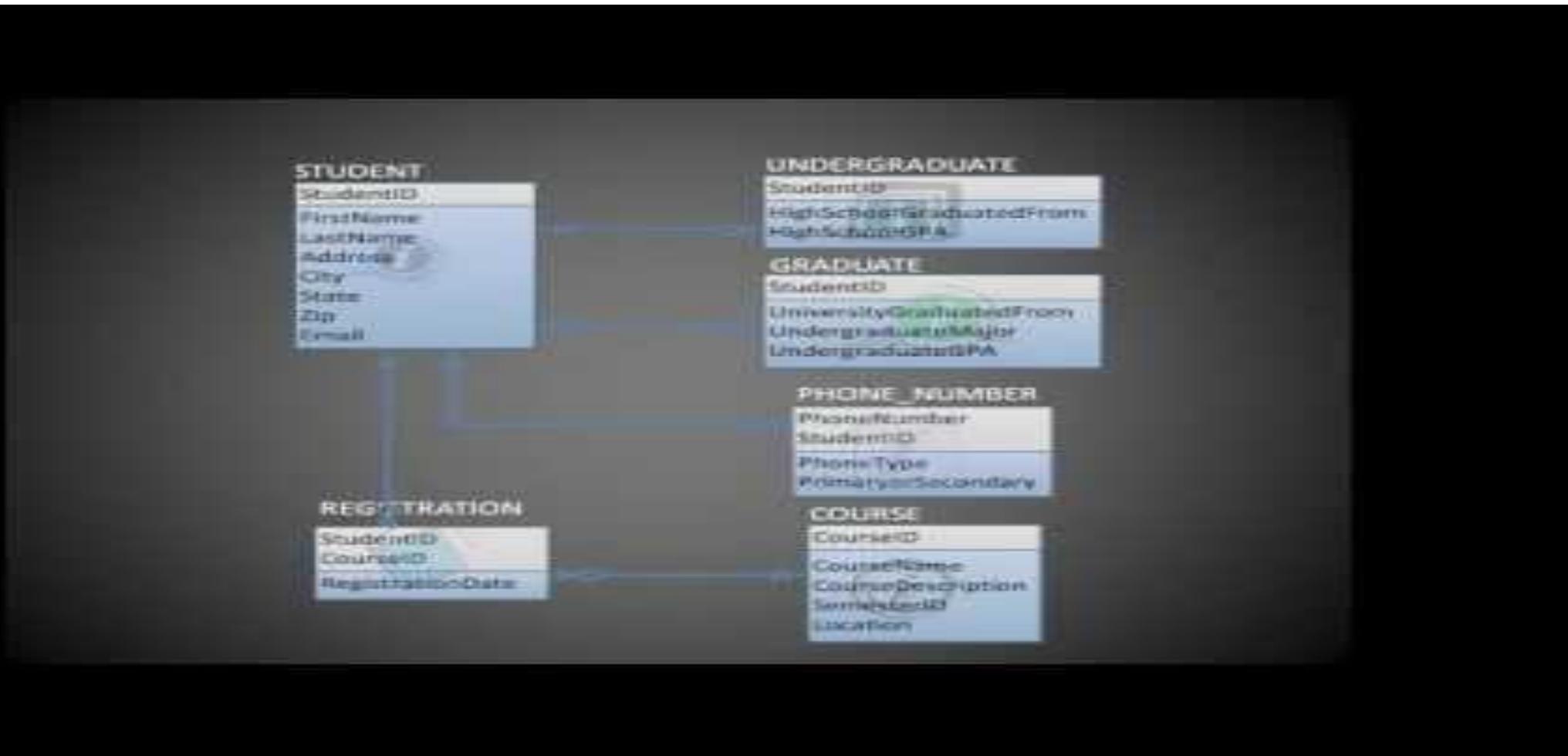
<https://youtu.be/QpdhBUYk7Kk>



Entity Relationship Diagrams.

Watch the video for the Entity Relationship Diagrams:

<https://youtu.be/-fQ-bRlhXc>



The normalisation process

Normalization is a database design technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization rules divides larger tables into smaller tables and links them using relationships. The purpose of Normalisation in SQL is to eliminate redundant (**repetitive**) data and ensure data is stored logically.

Database Normal Forms

Here is a list of Normal Forms in SQL:

1NF (First Normal Form)

2NF (Second Normal Form)

3NF (Third Normal Form)

First normal form (1NF): Entities do not contain repeated groups of attributes.

Second normal form (2NF): Entities are in 1NF and any non-key attributes depend upon the primary key. There are no partial dependencies.

Third normal form (3NF): Entities are in 2NF and all non-key attributes are independent. The table contains no non –key dependencies

The normalisation process

Database Normalization With Examples:

Database **Normalization Example** can be easily understood with the help of a case study. Assume, a video library maintains a database of movies rented out. Without any normalization in database, all information is stored in one table as shown below. Let's understand Normalization database with normalization example with solution

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3 rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5 th Avenue	Clash of the Titans	Mr.

Here you see **Movies Rented column has multiple values**. Now let's move into 1st Normal Forms

The normalisation process

First Normal Form (1NF) Rules:

- Each table cell should contain a single value.
- Each record needs to be unique.

The above table in 1NF-

1NF Example

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3 rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3 rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5 th Avenue	Clash of the Titans	Mr.

Example of 1NF in DBMS

The normalisation process

Before we proceed let's understand a few things.

What is a KEY in SQL?

A **KEY in SQL** is a value used to identify records in a table uniquely. An SQL KEY is a single column or combination of multiple columns used to uniquely identify rows or tuples in the table. SQL Key is used to identify duplicate information, and it also helps establish a relationship between multiple tables in the database.

Note: Columns in a table that are NOT used to identify a record uniquely are called non-key columns.

Database- Primary Key:

A primary is a single column value used to identify a database record uniquely.

It has following attributes

- A primary key cannot be NULL
- A **primary key** value must be unique
- The **primary key** values should rarely be changed
- The **primary key** must be given a value when a new record is inserted.



Primary Key

Primary Key in DBMS

The normalisation process

What is Composite Key?

A composite key is a primary key composed of multiple columns used to identify a record uniquely

In our database, we have two people with the same name Robert Phil, but they live in different places.

Composite Key

Robert Phil	3 rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5 th Avenue	Clash of the Titans	Mr.

Names are common. Hence you need name as well Address to uniquely identify a record.

Composite key in Database

Hence, we require both Full Name and Address to identify a record uniquely. That is a composite key.

Let's move into second normal form 2NF

The normalisation process

Second Normal Form (2NF) Rules:

- Rule 1- Be in 1NF
- Rule 2- Single Column Primary Key that does not functionally dependent on any subset of candidate key relation

It is clear that we can't move forward to make our simple database in 2nd Normalization form unless we partition the table above.

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr.

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

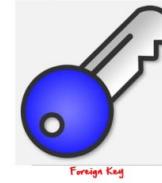
We have divided our 1NF table into two tables. Table 1 and Table2. Table 1 contains member information. Table 2 contains information on movies rented.

We have introduced a new column called Membership_ID which is the primary key for table 1. Records can be uniquely identified in Table 1 using membership id

The normalisation process

Database – Foreign Key:

In Table 2, Membership_ID is the Foreign Key

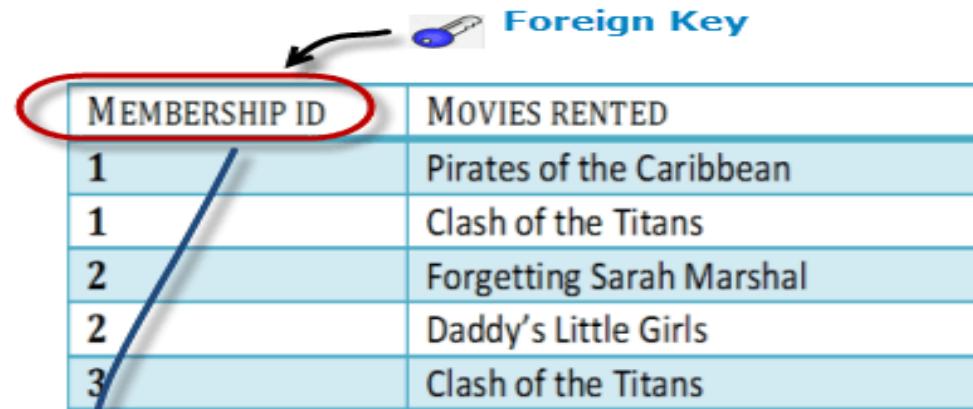


Foreign Key in DBMS

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Foreign Key references the primary key of another Table! It helps connect your Tables

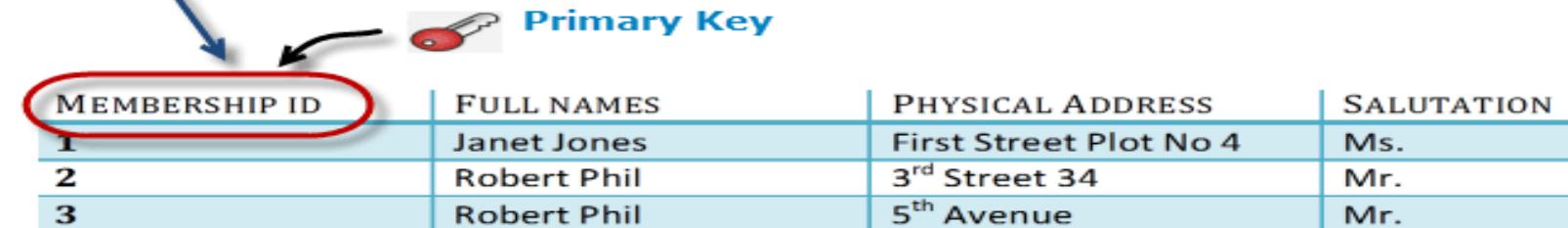
- A **foreign key** can have a different name from its **primary key**
- It ensures rows in one table have corresponding rows in another
- Unlike the **Primary key**, they do not have to be unique. Most often they aren't
- **Foreign keys** can be null even though **primary keys** can not



Foreign Key

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Foreign Key references Primary Key
Foreign Key can only have values present in primary key
It could have a name other than that of Primary Key



Primary Key

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr.

The normalisation process

Why do you need a foreign key?

Suppose, a novice inserts a record in Table B such as

Insert a record in Table 2 where Member ID =101

MEMBERSHIP ID	MOVIES RENTED
101	Mission Impossible

But Membership ID 101 is not present in Table 1

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr.

Database will throw an **ERROR**. This helps in referential integrity

You will only be able to insert values into your foreign key that exist in the unique key in the parent table. This helps in referential integrity.

The above problem can be overcome by declaring membership id from Table2 as foreign key of membership id from Table1

Now, if somebody tries to insert a value in the membership id field that does not exist in the parent table, an error will be shown!

The normalisation process

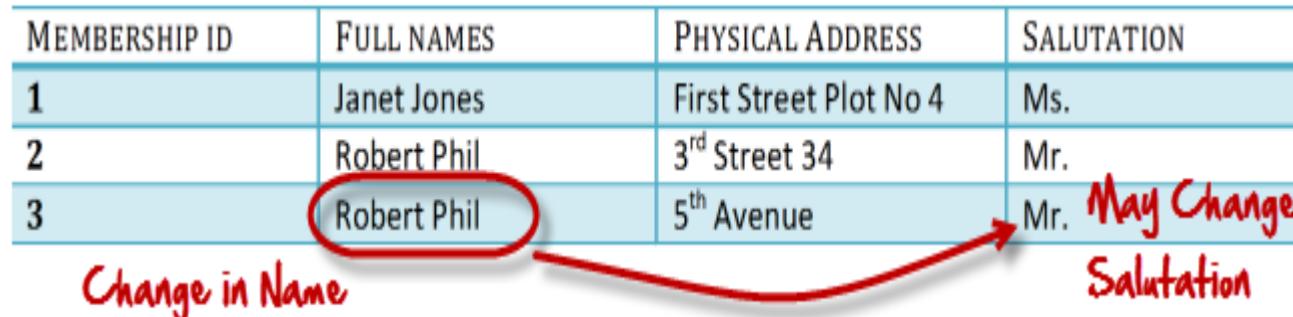
What are transitive functional dependencies?

A transitive functional dependency is when changing a non-key column, might cause any of the other non-key columns to change.

Consider the table 1. Changing the non-key column Full Name may change Salutation.

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 rd Street 34	Mr.
3	Robert Phil	5 th Avenue	Mr. <i>May Change</i>

Change in Name *Salutation*



Let's move into 3NF

The normalisation process

Third Normal Form (3NF) Rules:

- Rule 1- Be in 2NF
- Rule 2- Has no transitive functional dependencies

To move our 2NF table into 3NF, we again need to again divide our table.

3NF Example:

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3 rd Street 34	1
3	Robert Phil	5 th Avenue	1

Below is a 3NF example in SQL database:

We have again divided our tables and created a new table which stores Salutations.

There are no transitive functional dependencies, and hence our table is in 3NF

In Table 3 Salutation ID is primary key, and in Table 1 Salutation ID is foreign to primary key in Table 3

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

1st, 2nd and 3rd Normal Form (Database Normalisation)

Watch the video for the Normalization - 1NF, 2NF and 3NF :

<https://www.youtube.com/watch?v=J-drts33N8g>



Normalization - 1NF, 2NF, 3NF and 4NF

Watch the video for the Normalization - 1NF, 2NF, 3NF and 4NF:

<https://www.youtube.com/watch?v=UrYLYV7WSHM>

2nd Normal Form - All attributes (Non-Key Columns) dependent on the key

Primary Key	Cust ID	Cust Name	Shipping Address	Newsletter
st_smith	Alice Smith	35 Palm St, Miami	Xbox News	
roger123	Roger Banks	82 Campus Rd, Boston	PlayStation News	
wilson44	Evan Wilson	28 Rock Av, Denver	Xbox News	
wilson55	Evan Wilson	29 Rock Av, Denver	PlayStation News	
ste_smith	Alice Smith	42 Campus Rd, Boston	PlayStation News	

Primary Key	Item ID	Supplier	Supplier Phone	Price
1000	Xbox One	Sony	(800) 555-1234	250
1001	PlayStation 4	Sony	(800) 555-1234	300
1002	PS Vita	Sony	(800) 555-1234	200

Primary Key	Cust ID	Item ID
st_smith	Alice Smith	Xbox One
roger123	Roger Banks	PlayStation 4
wilson44	Evan Wilson	Xbox One
wilson55	Evan Wilson	PS Vita
ste_smith	Alice Smith	PlayStation 4

3rd Normal Form - All Fields (columns) can be determined Only by the Key in the table and no other column

More details about 1st Normal Form

All rows unique

Thanks to Learn Learn Scratch Tutorials for his work

First Normal Form

- All rows must be unique (no duplicate rows)
- Each cell must only contain a single value (not a list)
- Each value should be non-divisible (can't be split down further)

Problem 1 - All rows must be uniquely identifiable

Customer Name	Order
Bob Jones	Burger, Fries, Coke
Fred Jones	Nuggets, Lemonade, Fries
Bob Jones	Burger, Fries, Coke

Problem 1 - All rows must be uniquely identifiable

Customer Name	Order
Bob Jones	Burger, Fries, Coke
Fred Jones	Nuggets, Lemonade, Fries
Bob Jones	Burger, Fries, Coke



Identical Rows



Solution - add an order ID as a primary key

OrderID	Customer Name	Order
1	Bob Jones	Burger, Fries, Coke
2	Fred Jones	Nuggets, Lemonade, Fries
3	Bob Jones	Burger, Fries, Coke

Problem 2 - Each cell must only contain a single value

OrderID	Customer Name	Order
1	Bob Jones	Burger, Fries, Coke
2	Fred Jones	Nuggets, Lemonade, Fries
3	Bob Jones	Burger, Fries, Coke

Problem 2 - Each cell must only contain a single value

OrderID	Customer Name	Order
1	Bob Jones	Burger, Fries, Coke
2	Fred Jones	Nuggets, Lemonade, Fries
3	Bob Jones	Burger, Fries, Coke



Multiple Values!



Solution Create a separate table with order items

OrderID	Customer Name
1	Bob Jones
2	Fred Jones
3	Bob Jones

OrderID	Item
1	Burger
1	Fries
1	Coke
2	Nuggets
2	Lemonade
2	Fries
3	Burger
3	Fries
3	Coke

Problem 3 - All data must be atomic (non-divisible)

OrderID	Customer Name
1	Bob Jones
2	Fred Jones
3	Bob Jones

OrderID	Item
1	Burger
1	Fries
1	Coke
2	Nuggets
2	Lemonade
2	Fries
3	Burger
3	Fries
3	Coke

Problem 3 - All data must be atomic (non-divisible)

OrderID	Customer Name
1	Bob Jones
2	Fred Jones
3	Bob Jones

OrderID	Item
1	Burger
1	Fries
1	Coke
2	Nuggets
2	Lemonade
2	Fries
3	Burger
3	Fries
3	Coke

Solution

OrderID	First Name	Last Name
1	Bob	Jones
2	Fred	Jones
3	Bob	Jones

OrderID	Item
1	Burger
1	Fries
1	Coke
2	Nuggets
2	Lemonade
2	Fries
3	Burger
3	Fries
3	Coke

More details about 2nd Normal Form

No Partial Dependencies

Thanks to Learn Learn Scratch Tutorials for his work

Second Normal Form

- Database must be in First Normal Form
- Non partial dependency - All non-prime attributes should be fully functionally dependent on the candidate key

Example - No partial dependency

Student ID	Course ID	Course Fee
Composite Key		
1	1	500
1	2	1000
2	4	200
2	3	750
3	5	1000
3	3	750

This table is not in 2NF because the course Fee is only dependent on the primary key to determine the course fee, you can do it with just one column (course ID)

Example - No partial dependency

Student ID	Course ID	Course Fee
Composite Key		
1	1	500
1	2	1000
2	4	200
2	3	750
3	5	1000
3	3	750

This table is not in 2NF because the course Fee is only dependent on the primary key to determine the course fee, you can do it with just one column (course ID)

Example - No partial dependency

Student Courses

Student ID	Course ID
Composite Key	
1	1
1	2
2	1
2	3
3	5
3	3

Course Fees

Course ID	Course Fee
1	500
2	1000
3	750
4	200
5	1000
6	300

More details about 3rd Normal Form (3NF)

No Transitive Dependencies

Thanks to Learn Learn Scratch Tutorials for his work

Third Normal Form

- Database must be in First & Second Normal Form
- No transitive dependency - All fields must only be determinable by the primary/composite key, not by other keys

Third Normal Form

Composite Key

Tournament Name	Year	Winner	Winner's DOB
Indiana Invitational	1998	Al Fredrickson	21 July 1975
Cleveland Open	1999	Bob Albertson	28 September 1968
Des Moines Masters	1999	Al Fredrickson	21 July 1975
Indiana Invitational	1999	Chip Masterson	14 March 1977

Third Normal Form

Composite Key

Tournament Name	Year	Winner	Winner's DOB
Indiana Invitational	1998	Al Fredrickson	21 July 1975
Cleveland Open	1999	Bob Albertson	28 September 1968
Des Moines Masters	1999	Al Fredrickson	21 July 1975
Indiana Invitational	1999	Chip Masterson	14 March 1977

Third Normal Form

Tournament Winners

Composite Key

<u>Tournament</u>	<u>Year</u>	Winner
Indiana Invitational	1998	Al Fredrickson
Cleveland Open	1999	Bob Albertson
Des Moines Masters	1999	Al Fredrickson
Indiana Invitational	1999	Chip Masterson

Winners DOBs

Winner	Date of birth
Chip Masterson	14 March 1977
Al Fredrickson	21 July 1975
Bob Albertson	28 September 1968

Database management systems DBMS

A database management system is used to organise who can access a database and how they can make changes. They are important for managing the security and the integrity of data.

A **database management system (DBMS)** is a tool to store, edit and organise data in a **database**. It provides several key features:

- stores data in one central location
- allows data to be shared by many users
- provides **user interfaces** to work with the data
- creates **backups**
- controls who can access and edit the data

Benefits of a DBMS

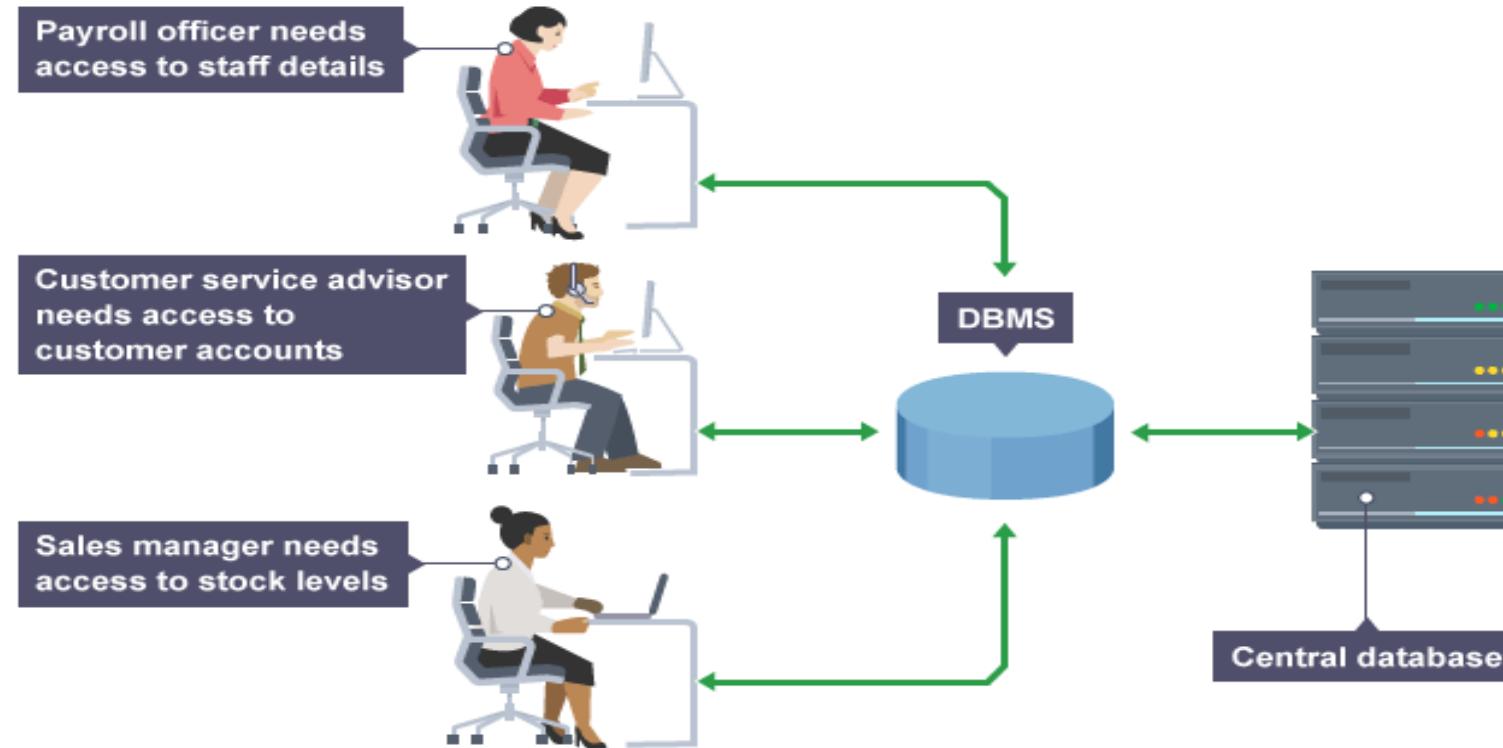
The benefits of a DBMS include:

- **integrity** - the structure of the database can change, but the **applications** using the data do not need to be changed
- **efficiency** - avoids data duplication and inconsistency, and less storage space is taken up because data is shared
- **consistency** - data is the same, regardless of who is viewing it
- **backups** - it is easy to back up data from one location
- **security** - the data is in a secure central place and different access rights can be assigned to different people
- **customisation** - applications can be customised to suit the needs of the user

Database management systems DBMS

An important part of a DBMS is separating applications from the data. When people use the applications they call on the data they need to work on. They do not need to use all the data every time they use the database.

A database in a DBMS could be viewed by lots of different people with different responsibilities



Picture Courtesy of BBC

For example, within a company there are different departments, as well as customers, who each need to see different kinds of data. Each employee in the company will have different levels of access to the database with their own customised **front-end** application

Table Source: https://en.wikipedia.org/wiki/Third_normal_form

Database management systems DBMS

An important part of a DBMS is separating applications from the data. When people use the applications they call on the data they need to work on. They do not need to use all the data every time they use the database.

A database in a DBMS could be viewed by lots of different people with different responsibilities

DBMS example:

The following example shows how a **DBMS** is used for a ticketing website.

A ticketing company allows customers to buy tickets online or through a booking office. **How can they use a DBMS to keep a record of sales?**

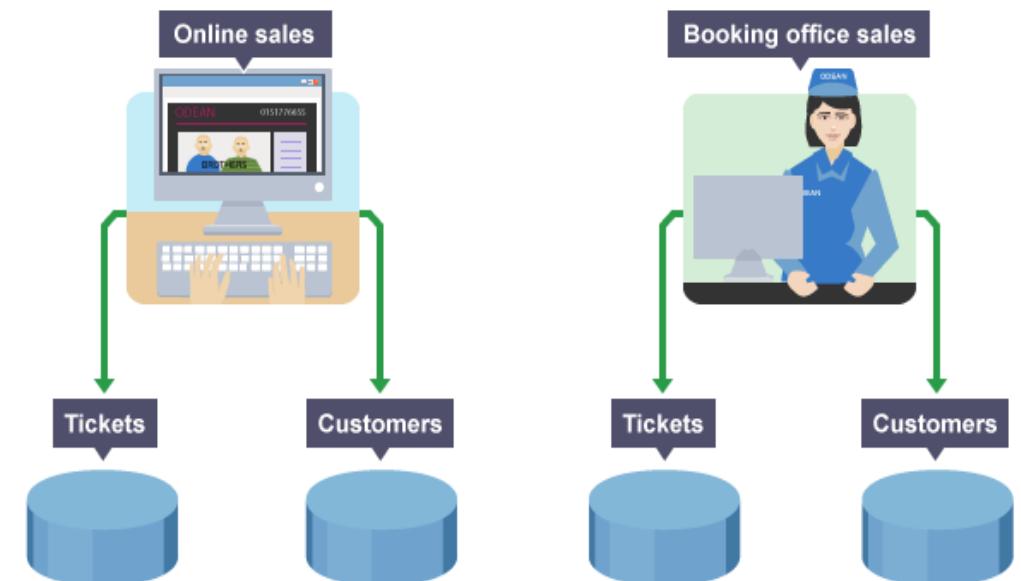
Option 1

They could have two separate **databases** - one for online sales and one for booking office sales. But they would both need to keep records of tickets and customers.

This type of system would **not** work effectively for a number of reasons:

- How would you know which seats had been sold in each system?
- Would you divide the seats up before offering them to the public?
- What would happen if online tickets sold out before booking office tickets did?

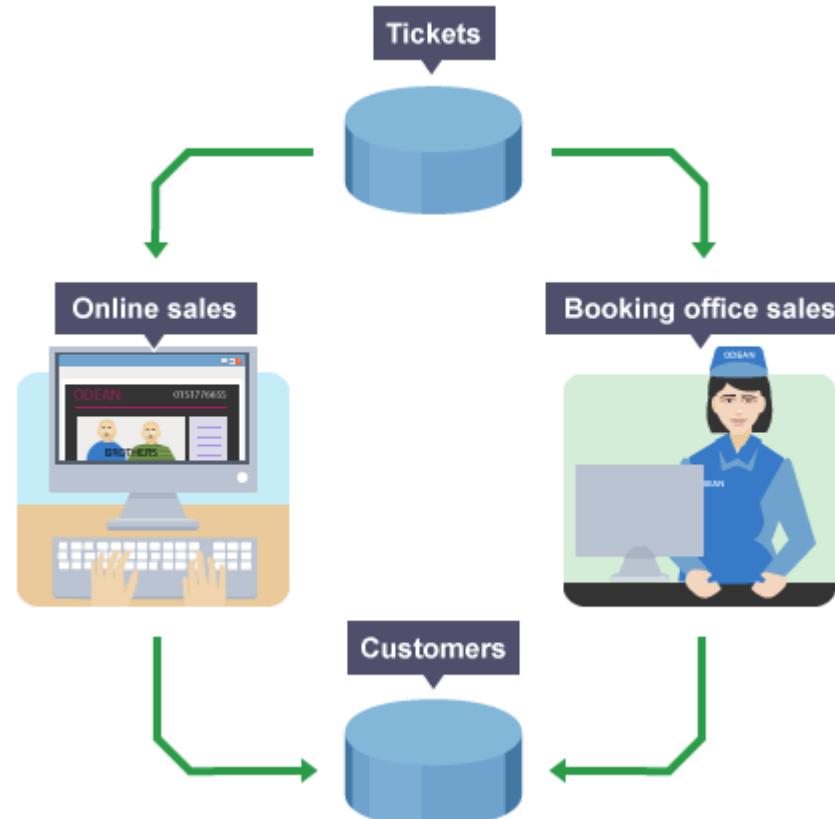
You could lose customers due to errors or double-selling the same tickets.



Database management systems DBMS

Option 2:

A DBMS could create a system where the ticket and customer data could be held in one database. There could be an **application** for the booking office and the online sales which would allow both of them to access the database. Data could then be shared between online sales and booking office sales and the data would not be duplicated.



Picture Courtesy of BBC

This type of system would work much more effectively.

Database management systems DBMS

Database views

One of the benefits of using a **DBMS** is that it allows different views of a **database**. This is useful as different users have different requirements.

The **external** view is the view of the data used by customers or people who input and view data, a **conceptual** view is useful for the database designer and the **internal** view is used for the detailed programming of the database.

A customer does not need to see as much information as the administrator or programmer of the database.

External view

The external view is the view that the **customer** would see. If you made an **app** which used a database to collect data, you might want to give users a simple form to add new information to the app. For example, if there was an app which people used for bird watching it could simply have a form which includes:

- bird type
- time of year
- date
- location

The app user does not need to see the whole database to add their information. It is possible for a database user to work with a database and never actually see the **tables** that are used to store the data.

Conceptual view

This view shows how the tables are all connected. It could be used to plan changes to the database. This view can also be presented using **SQL**.

Database management systems DBMS

Internal view

This is the view that shows the detail of how the computer sees the database. It might include details of how files are structured and stored and include **hex** or **binary** references to **bytes** of data. It is complex and would only be used by the database programmer.

Access levels

One of the challenges of creating a **DBMS** is managing who can access and change the data.

If anyone can edit the database, data could easily become **corrupted**. It is better to have different levels of access for the database to maintain the **integrity** of the database.

For example, with personalised websites that require a login, you will only see data that is relevant to you. However, if an employee of that website company accesses the DBMS, they will be able to view all customer accounts. The customer details come from the same database, but different levels of access are given to different users.

A DBMS developer would think about the needs of the user and develop a bespoke system to suit the needs of the company.

With an online music library, there would be different access levels for different users. For example:

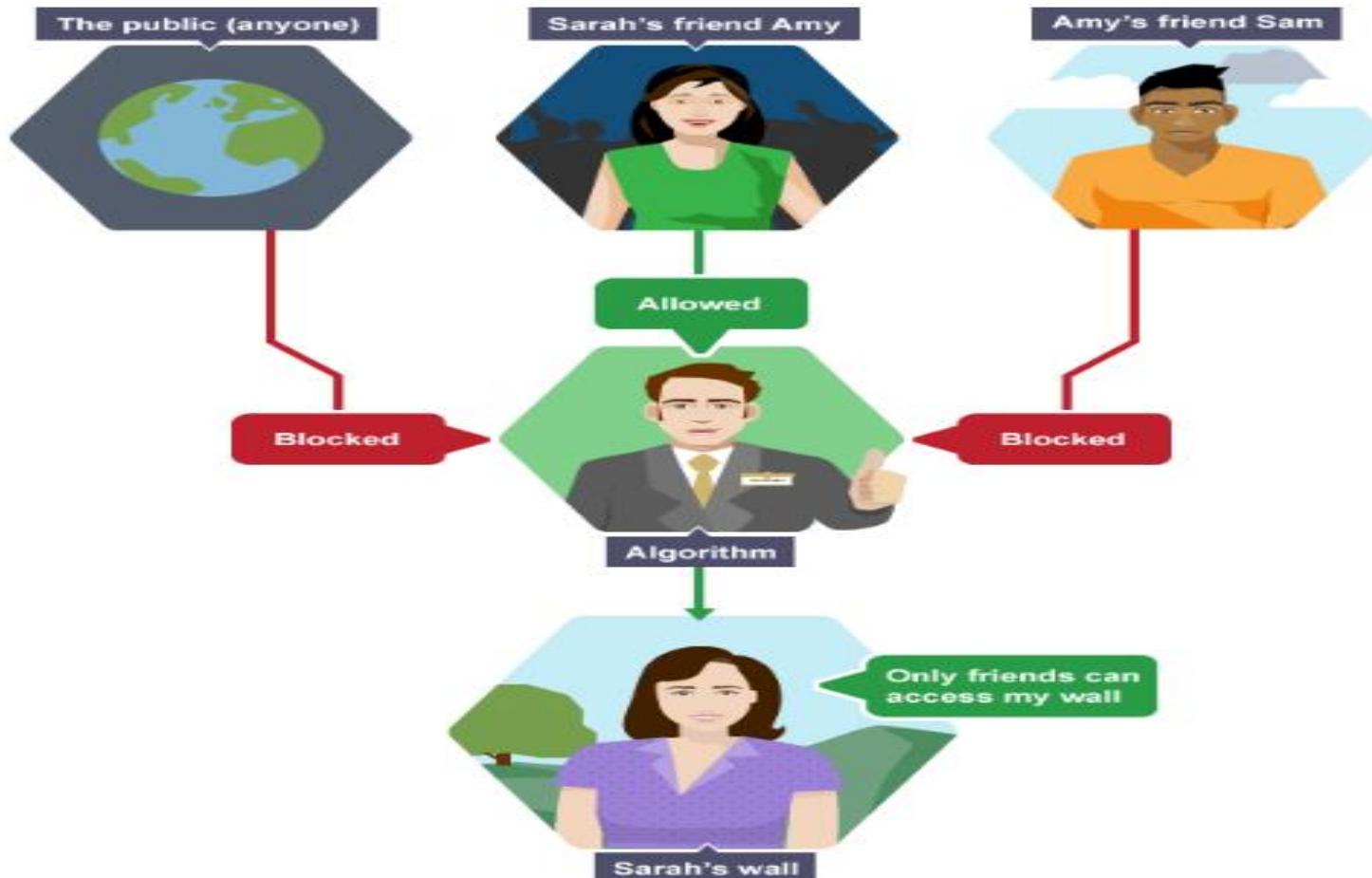
- **customers** should only see their own music
- **administrators** would have access to **upload** and amend the entire library
- **account managers** would be able to see the financial details for the customers when required

Websites use database applications to customise the user interface to suit the needs of each user. The owner of a **blog** will be able to delete, add and edit comments, whereas the reader would simply be able to view the blog posts.

Database management systems DBMS

With social **networks** you customise who can access your data:

- you might only allow 'friends' to see your data
- you might allow 'friends' and 'friends of friends' to see your data
- you might have a public account that is visible to everyone



Picture Courtesy of BBC

Database management systems DBMS

[Watch the video for the Data Redundancy and Data Anomalies](#)

<https://www.youtube.com/watch?v=KgbsIENXyK4>



Database management systems DBMS

The data that appears on Sarah's profile on a social network is stored in a database. Sarah has set up access rights to her data. Sarah only allows users known as her 'friends' to view her profile. This means that the public and 'friends of friends' cannot see her data.

Concurrency and ACID

Concurrency is when many users are interacting with a system or many changes are being made at once.

A **DBMS** needs to use concurrency. However, the **database** needs to prevent two people using the data in a conflicting way. For example, two people visiting an online shop might try to buy the final product in store. The database would need to make sure only one person could buy it.

ACID rules

A change in a database is called a **transaction**. Changes to databases must conform to ACID rules:

- **Atomicity** - the transaction must be completed fully. If it is not completed fully it will not be recorded.
- **Consistency** - any change must not break the database. It must be consistent with how it was before the change.
- **Isolation** - a transaction must be isolated and not interfere with another transaction.
- **Durability** - a transaction must remain in the database.

Database management systems DBMS

Advantage of DBMS

There are several advantages of Database management system over file system. Few of them are as follows:

No redundant data: Redundancy removed by data [normalization](#). No data duplication saves storage and improves access time.

Data Consistency and Integrity: As we discussed earlier the root cause of data inconsistency is data redundancy, since data normalization takes care of the data redundancy, data inconsistency also been taken care of as part of it

Data Security: It is easier to apply access constraints in database systems so that only authorized user is able to access the data. Each user has a different set of access thus data is secured from the issues such as identity theft, data leaks and misuse of data.

Privacy: Limited access means privacy of data.

Easy access to data – Database systems manages data in such a way so that the data is easily accessible with fast response times.

Easy recovery: Since database systems keeps the backup of data, it is easier to do a full recovery of data in case of a failure.

Flexible: Database systems are more flexible than file processing systems.

Database management systems DBMS

Drawbacks (Disadvantage)

Data redundancy: Data redundancy occurs when the same data point is multiplied across the database and can be found repeated in an unnecessary form. Data redundancy refers to the duplication of data. A common example of data redundancy is when a name and address are both present in different columns within a table. If the link between these data points is defined in every single new database entry it would lead to unnecessary duplication across the entire table.

Data inconsistency: Data inconsistency is a situation where there are multiple tables within a database that deal with the same data but may receive it from different inputs. Example – If we have an address of someone in many tables and when we change it in only one table and in another table it may not be updated so there is the problem of data inconsistency may occur.

Data Isolation: Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.

Dependency on application programs: Changing files would lead to change in application programs.

Atomicity issues: Atomicity of a transaction refers to “All or nothing”, which means either all the operations in a transaction executes or none. For example: Let's say Steve transfers 100\$ to Negan's account. This transaction consists multiple operations such as debit 100\$ from Steve's account, credit 100\$ to Negan's account. Like any other device, a computer system can fail let's say it fails after first operation then in that case Steve's account would have been debited by 100\$ but the amount was not credited to Negan's account, in such case the rollback of operation should occur to maintain the atomicity of transaction. It is **difficult to achieve atomicity in file processing systems.**

Data Security: Data should be secured from unauthorized access, for example a student in a college should not be able to see the payroll details of the teachers, such kind of security constraints are difficult to apply in file processing systems.

Database management systems DBMS

- State what is meant by the term data redundancy.

Data redundancy: Repeated / duplicated data

- Explain how a relational database can help to reduce data redundancy.

- Because each record/piece of data is stored once and is referenced by a (primary) key
- Because data is stored in individual tables and the tables are linked by relationships
- By the proper use of Primary and Foreign keys
- By enforcing referential integrity
- By going through the normalization process

- Explain the difference between security and integrity.

- Security ensures that data is safe from unauthorized access // safe from loss
- Integrity ensures that data is accurate / consistent / up to date

- Name and describe two security features provided by a DBMS.

For example:

- Access rights // User accounts
- Restrict actions (e.g. read / read-write) of specific users // unauthorized users cannot access the database
- Views
- Restrict which parts of the database specific users can see
- Password // Biometrics // PIN code
- Prevents unauthorised access
- Automatic Backup
- Create regular copies of data in case of loss
- Encryption
- Data is incomprehensible to unauthorized users

Database management systems DBMS

- The DBMS has a data dictionary. **Describe what the data dictionary stores.**
 - Stores all the information about the database // data about the data // metadata about the data
 - For example, fields, data types, validation, keys
- The DBMS has a query processor. **Describe the purpose of a query processor.**
 - Allows the user to enter criteria
 - Searches for data which meets the entered criteria
 - Organizes the results to be displayed to the user
- Relationships are created between tables using primary and foreign keys. **Describe the role of a primary and a foreign key in database relationships.**
 - Primary key uniquely identifies each tuple // Each tuple in the table is unique
 - Primary key can be used as a foreign key in another table to form a link/relationship between the tables

By example:

- Identification of a primary key in a table
- Describing that primary key in another table as a foreign key
- A database administrator decides to enforce referential integrity. **What is meant by referential integrity.**
 - Referential integrity is making sure tables do not try to reference data which does not exist // A value of one attribute of a table exists as a value of another attribute in a different table
 - A primary key cannot be deleted unless all dependent records are already deleted
 - Cascading delete
 - A primary key cannot be updated unless all dependent records are already updated
 - Cascading update / edit
 - Every foreign key value has a matching value in the corresponding primary key
 - The foreign keys must be the same data type as the corresponding primary key

Learning Objectives

• 8.3 Data Definition Language (DDL) and Data Manipulation Language (DML)

- Show understanding that DBMS carries out all creation / modification of the database structure using its Data Definition Language (DDL)
- Show understanding that the DBMS carries out all queries and maintenance of data using its DML
- Show understanding that the industry standard for both DDL and DML is Structured Query Language (SQL): Understand a given SQL script
- Understand given SQL (DDL) commands and be able to write simple SQL (DDL) commands using a sub-set of commands
 - ❖ Create a database (CREATE DATABASE)
 - ❖ Create a table definition (CREATE TABLE), including the creation of attributes with appropriate data types:
CHARACTER, VARCHAR(n), BOOLEAN, INTEGER, REAL, DATE, TIME
 - ❖ change a table definition (ALTER TABLE)
 - ❖ add a primary key to a table (PRIMARY KEY (field))
 - ❖ add a foreign key to a table (FOREIGN KEY (field) REFERENCES Table (Field))
- Write an SQL script to query or modify data (DML) which are stored in (at most two) database tables :
 - ❖ Queries including SELECT... FROM, WHERE, ORDER BY, GROUP BY, INNER JOIN, SUM, COUNT, AVG
 - ❖ Data maintenance including. INSERT INTO, DELETE FROM, UPDATE

Database management systems DBMS

- **Example:** referential integrity.
 - A UserName cannot be deleted from the USER table if they have a related photo/textpost
 - If UserName is updated in USER table, it must also be updated in PHOTO and TEXTPOST tables
 - Cannot create/edit a record in TEXTPOST / PHOTO without a matching entry in USER table
- **Define the three stages of database normalisation.**
 - 1NF: No repeated groups of attributes and All attributes should be atomic also No duplicate rows
 - 2NF: in 1NF and No partial dependencies
 - 3NF: in 2NF and No non-key dependencies and No transitive dependencies
- The database manager is concerned about data integrity. **State what is meant by data integrity. Give an example of how the manager can ensure data integrity in database.**

Ensure data is consistent / accurate // keep data consistent / accurate
for example from:

 - Validation rules, • Referential integrity , • Verification , • Input masks , • Setting data types
 - Removing redundant data
 - Backup data
 - Access controls
 - Audit trail

Data Definition Language (DDL) and Data Manipulation Language(DML)

DDL:

DDL is Data Definition Language which is used to define data structures. For example: create table, modify (Alter) and remove table in the data structures that form a relational database or in the instructions in SQL. (CREATE, ALTER , DROP)

DML:

DML is Data Manipulation Language which is used to manipulate data itself. For example: insert (add), update (modify), delete and retrieve the data stored in the a relational database or in the instructions in SQL. (SELECT FROM, WHERE, ORDER BY, GROUP BY, INNER JOIN, SUM, COUNT, AVG)

Why DDL:

Here, are reasons for using DDL method:

- Allows you to store shared data
- Data independence improved integrity
- Allows multiple users
- Improved security efficient data access

Why DML:

Here, are reasons for using DML method:

- The DML statements allow you to modify the data stored in a database.
- Users can specify what data is needed.
- DML offers many different flavors and capabilities between database vendors.
- It offers an efficient human interaction with the system.

Data Definition Language (DDL) and Data Manipulation Language(DML)

Difference Between DDL and DML in DBMS

DDL	DML
It stands for Data Definition Language.	It stands for Data Manipulation Language.
It is used to create database schema and can be used to define some constraints as well.	It is used to add, retrieve or update the data.
It basically defines the column (Attributes) of the table.	It add or update the row of the table. These rows are called as tuple.
It doesn't have any further classification.	It is further classified into Procedural and Non-Procedural DML.
Basic command present in DDL are CREATE, DROP, RENAME, ALTER etc.	BASIC command present in DML are UPDATE, INSERT, MERGE etc.
DDL does not use WHERE clause in its statement.	While DML uses WHERE clause in its statement.

Data Definition Language (DDL) and Data Manipulation Language(DML)

DDL Commands:

The Data Definition Languages (DDL) Commands are as follows

CREATE TABLE– It is used to create a new table or a new database.

ALTER TABLE– It is used to alter or change the structure of the database table.

DROP TABLE – It is used to delete a table, index, or views from the database.

TRUNCATE – It is used to delete the records or data from the table, but its structure remains as it is.

RENAME – It is used to rename an object from the database.

DML Commands:

DML in the Database Management System (DBMS) are as follows:

•**SELECT FROM** – Retrieve data from the database.

•**INSERT INTO** – Insert data into a table.

•**UPDATE** – Update existing data within a table.

•**DELETE FROM** – Delete records from a database table.

•**WHERE:** Includes only row a database, Queries that match a given condition

•**ORDER BY:** Sorts the result from a query by a given column either alphabetically or numerically

•**GROUP BY:** Arranges data into groups

•**INNER JOIN:** Combines row from different tables if the join condition is true

•**SUM:** Returns the sum of all the values in the column

•**COUNT:** Counts the number of rows where the column is not NUL

•**AVG:** Returns the average value for a column with a numeric data type

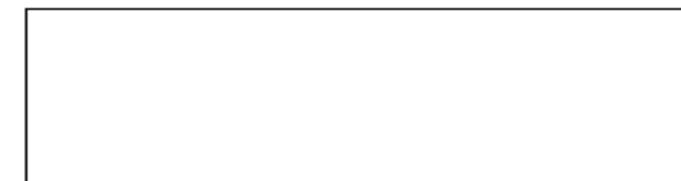
AS/A Level Computer Science Exam Question 1:

Mehrdad has a holiday company database that includes:

- data about holidays, such as the location, date, duration (in days)
- data about the customers and the holidays they have booked.

(a) Mehrdad has **normalised** the database, which has three tables.

(i) Draw an entity–relationship (E–R) diagram for the **normalised** tables.



[3]

AS/A Level Computer Science Exam Question 1:

- (ii) Complete the table to identify the primary key and foreign key(s) for each of the tables you identified in part (a)(i). If the table has no foreign key(s), write 'None'.

Table name	Primary key	Foreign key

[3]

- (iii) Explain why the holiday database is in Third Normal Form (3NF).

.....

.....

.....

.....

[2]

AS/A Level Computer Science Exam Question 1:

- (b) The holiday company has several members of staff. The database has two additional tables to store data about the staff.

STAFF (StaffID, FirstName, SecondName, DateOfBirth, Role, Salary)

SCHEDULE (ScheduleID, StaffID, WorkDate, Morning, Afternoon)

The following table shows some sample data from the table SCHEDULE.

ScheduleID	StaffID	WorkDate	Morning	Afternoon
210520-1	BC	21/05/2020	TRUE	TRUE
210520-2	JB	21/05/2020	TRUE	FALSE
220520-1	BC	22/05/2020	FALSE	TRUE
220520-2	LK	22/05/2020	TRUE	FALSE

- (i) Write an SQL script to display the first name and second name of all staff members working on 22/05/2020.

.....
.....
.....
.....

AS/A Level Computer Science Exam Question 1:

- (ii) Write an SQL script to count the number of people working on the morning of 26/05/2020.

.....

.....

.....

.....

.....

.....

..... [3]

AS/A Level Computer Science Exam Answer 1:

Question	Answer	Marks	Guidance												
3(a)(i)	<p>1 mark per bullet</p> <ul style="list-style-type: none">• 3 suitable names• 1 Customer can have many Bookings• 1 Holiday can have many Bookings <p>e.g.</p> <pre>classDiagram class Customer class Holiday class Booking Customer "2..>" Holiday</pre>	3	0 marks for a many-to-many relationship between Customer and Holiday. Accept any recognised method of 1-to-many												
3(a)(ii)	<p>1 mark for 3 appropriate Primary Keys 1 mark for None in Customer and Holiday 1 mark for two FKs in booking that match the PKs in Customer and Holiday</p> <table border="1"><thead><tr><th>Table Name</th><th>Primary Key</th><th>Foreign Key</th></tr></thead><tbody><tr><td>Customer</td><td>CustomerID</td><td>None</td></tr><tr><td>Booking</td><td>BookingID</td><td>CustomerID HolidayID</td></tr><tr><td>Holiday</td><td>HolidayID</td><td>None</td></tr></tbody></table>	Table Name	Primary Key	Foreign Key	Customer	CustomerID	None	Booking	BookingID	CustomerID HolidayID	Holiday	HolidayID	None	3	Allow FT in names and structure
Table Name	Primary Key	Foreign Key													
Customer	CustomerID	None													
Booking	BookingID	CustomerID HolidayID													
Holiday	HolidayID	None													

AS/A Level Computer Science Exam Answer 1:

Question	Answer	Marks	Guidance
3(a)(iii)	<p>1 mark per bullet</p> <ul style="list-style-type: none">• No many-to-many relationships // only two 1-many relationships• All fields in each table are fully dependant on the PKs for each table	2	
3(b)(i)	<p>1 mark per bullet</p> <ul style="list-style-type: none">• Selecting First name and Second name• From staff (and schedule)• Joining tables (inner join, or AND statement)• WHERE SCHEDULE.WorkDate = '22/5/2020' <p>e.g.</p> <pre>SELECT STAFF.FirstName, STAFF.SecondName FROM STAFF, SCHEDULE WHERE SCHEDULE.WorkDate = '22/05/2020' AND SCHEDULE.StaffID = STAFF.StaffID;</pre>	4	
3(b)(ii)	<p>1 mark per bullet</p> <ul style="list-style-type: none">• Selecting and using count on a field• From the table schedule• Where WorkDate = 26/5/2020 and Morning = TRUE <p>e.g.</p> <pre>SELECT COUNT(StaffID) FROM SCHEDULE WHERE WorkDate = '26/05/2020' AND Morning = TRUE;</pre>	3	

AS/A Level Computer Science Exam Question 2:

A software company produces software and distributes it under different software licences.

- (a)** Four descriptions of software licences are given.

Write the type of software licence that best fits each description. Use a different type of licence for each description.

1. The software can be legally used, only after a fee has been paid.

Licence type

2. The source code comes with the software. If the software is modified, the edited source code must be released under the same conditions as the original software.

Licence type

3. The software is free for a trial period and then a fee is requested, or expected, if the user wants to continue to use the software.

Licence type

4. The source code comes with the software. The software is free to be downloaded, edited, and distributed, possibly without restriction.

Licence type

[4]

AS/A Level Computer Science Exam Question 2:

(b) The software company stores information about customers and the software licences they have purchased. The company considers a file-based approach for the storage and retrieval of data.

(i) Give **three** limitations of a file-based approach to store the data.

1

.....

2

.....

3

.....

[3]

(ii) The software company decides to use a database to overcome the limitations of a file-based system. Some of these limitations are addressed through the logical schema.

Name **and** describe **two** levels of the schema of a database.

Name 1

Description

.....

Name 2

Description

.....

[4]

AS/A Level Computer Science Exam Question 2:

- (c)** The database has the following tables:

CUSTOMER (CustomerID, CompanyName)

SOFTWARE (SoftwareID, SoftwareName, OperatingSystem, Description)

LICENCE (LicenceID, CustomerID, SoftwareID, DateOfPurchase,
LicenceType, Cost, ExpiryDate)

- (i)** Identify the type of relationship that exists between the tables CUSTOMER and LICENCE.

.....
..... [1]

- (ii)** Describe how the relationship is created between the tables CUSTOMER and LICENCE.

.....
.....
.....
..... [2]

AS/A Level Computer Science Exam Question 2:

- (iii) The company needs a list of all software licences that have an expiry date on or before 31/12/2019.

Write an SQL query to return the fields CustomerID, SoftwareID, LicenceType, Cost and ExpiryDate for all licences that expire on, or before 31/12/2019. Group the output by CustomerID, and in ascending order of cost.

[5]

[5]

AS/A Level Computer Science Exam Answer 2:

Question	Answer	Marks
2(a)	<p>1 mark for each correct term</p> <ul style="list-style-type: none"><input type="checkbox"/> Commercial Licence<input type="checkbox"/> Free Software Licence<input type="checkbox"/> Shareware Licence<input type="checkbox"/> Open Source Licence	4
2(b)(i)	<p>1 mark per bullet point to max 3</p> <ul style="list-style-type: none"><input type="checkbox"/> Data redundancy // data is repeated in more than one file<input type="checkbox"/> Data dependency // changes to data means changes to programs accessing that data<input type="checkbox"/> Lack of data integrity // entries that should be the same can be different in different places<input type="checkbox"/> Lack of data privacy // all users have access to all data if a single flat file	3

AS/A Level Computer Science Exam Answer 2:

2(b)(ii)	<p>1 mark for each correct name, 1 mark for each matching description, max 2 marks per level</p> <ul style="list-style-type: none"><input type="checkbox"/> External<input type="checkbox"/> The individual's view(s) of the database<input type="checkbox"/> Conceptual<input type="checkbox"/> Describes the data as seen by the applications making use of the DBMS<input type="checkbox"/> Describes the 'views' which users of the database might have<input type="checkbox"/> Physical / Internal<input type="checkbox"/> Describes how the data will be stored on the physical media<input type="checkbox"/> Logical<input type="checkbox"/> Describes how the relationships will be implemented in the logical structure of the database	4
2(c)(i)	1-to-many // 1 customer to/has many licences	1
2(c)(ii)	<p>1 mark per bullet point</p> <ul style="list-style-type: none"><input type="checkbox"/> <u>CustomerID</u> is the Primary key in <u>CUSTOMER</u> table<input type="checkbox"/> Links to <u>CustomerID</u> as a Foreign key in <u>LICENCE</u> table	2

AS/A Level Computer Science Exam Answer 2:

Question	Answer	Marks
2(c)(iii)	<p>1 mark per bullet point</p> <ul style="list-style-type: none"><input type="checkbox"/> Select with correct 5 fields<input type="checkbox"/> From LICENCE<input type="checkbox"/> Where ExpiryDate <= '31/12/2019' (any appropriate date type)<input type="checkbox"/> Group by CustomerID<input type="checkbox"/> Order by Cost (with or without ASC, but not DESC) <pre>SELECT CustomerID, SoftwareID, LicenceType, Cost, ExpiryDate FROM LICENCE WHERE ExpiryDate <= '31/12/2019' GROUP BY CustomerID ORDER BY Cost;</pre>	5

AS/A Level Computer Science Exam Question 3:

- Moheem is creating a relational database to store data about his customers.
 - (a) Moheem has been told a relational database addresses some of the limitations of a file-based approach by reducing data redundancy.
 - (i) State what is meant by the term **data redundancy**.

.....
..... [1]

- (ii) Explain **how** a relational database can help to reduce data redundancy.

.....
.....
.....
.....
..... [3]

AS/A Level Computer Science Exam Question 3:

(b) Moheem uses a Database Management System (DBMS) to ensure the security and integrity of the data.

(i) Explain the difference between security and integrity.

.....
.....
.....
..... [2]

(ii) Name **and** describe **two** security features provided by a DBMS.

Feature 1

.....
.....
.....

Feature 2

.....
.....

AS/A Level Computer Science Exam Question 3:

- (iii) The DBMS provides software tools for the database developer.

Fill in the names of the missing software tools in the following statements.

A allows a developer to extract data from a database.

A enables a developer to create user-friendly forms and reports.

[2]

AS/A Level Computer Science Exam Answer 3:

Question	Answer	Marks
5(a)(i)	1 mark for correct answer <u>Repeated / duplicated data</u>	1
5(a)(ii)	1 mark per bullet point <ul style="list-style-type: none">• Because each record/piece of data is stored once <u>and</u> is referenced by a (primary) key• Because data is stored in individual tables• ...and the tables are linked by relationships• By the proper use of Primary and Foreign keys• By enforcing referential integrity• By going through the normalisation process	3
5(b)(i)	1 mark per bullet point <ul style="list-style-type: none">• Security ensures that data is safe from unauthorised access // safe from loss• Integrity ensures that data is accurate / consistent / up to date	2

AS/A Level Computer Science Exam Answer 3:

Question	Answer	Marks
5(b)(ii)	<p>1 mark for naming, 1 mark for description</p> <p>For example:</p> <ul style="list-style-type: none">• Access rights // User accounts• Restrict actions (e.g. read / read-write) of specific users // unauthorised users cannot access the database• Views• Restrict which parts of the database specific users can see• Password // Biometrics // PIN code• Prevents unauthorised access• Automatic Backup• Create regular copies of data in case of loss• Encryption• Data is incomprehensible to unauthorised users	4
5(b)(iii)	<p>1 mark per bullet</p> <ul style="list-style-type: none">• Query Processor• Developer Interface	2

AS/A Level Computer Science Exam Question 4:

- A company uses a relational database, EMPLOYEES, to store data about its employees and departments.

(a) The company uses a Database Management System (DBMS).

(i) The DBMS has a data dictionary.

Describe what the data dictionary stores.

.....
.....
.....
.....

[2]

(ii) The DBMS has a query processor.

Describe the purpose of a query processor.

.....
.....
.....
.....

[2]

AS/A Level Computer Science Exam Question 4:

- (b) Relationships are created between tables using primary and foreign keys.

Describe the role of a primary and a foreign key in database relationships.

.....

.....

.....

..... [2]

AS/A Level Computer Science Exam Question 4:

(c) In the company:

- An employee can be a manager.
- A department can have several managers and several employees.
- An employee can only belong to one department.

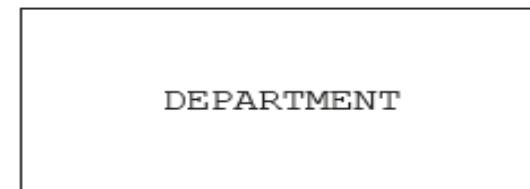
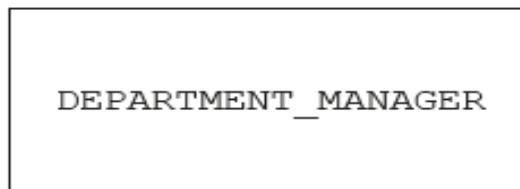
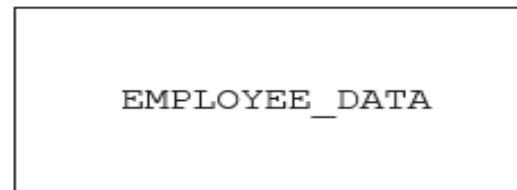
The EMPLOYEES database has three tables:

EMPLOYEE_DATA (EmployeeID, FirstName, LastName, DateOfBirth, Gender,
DepartmentNumber)

DEPARTMENT (DepartmentNumber, DepartmentName)

DEPARTMENT_MANAGER (DepartmentNumber, EmployeeID, role)

Complete the entity-relationship (E-R) diagram for the EMPLOYEES database.



AS/A Level Computer Science Exam Question 4:

(d) Give **three** reasons why the EMPLOYEES database is fully normalised.

1

2

3

[3]

AS/A Level Computer Science Exam Question 4:

(e) Part of the EMPLOYEE_DATA table is shown.

EmployeeID	FirstName	LastName	DateOfBirth	Gender	DepartmentNumber
156FJEK	Harvey	Kim	12/05/1984	Male	S1
558RRKL	Catriona	Moore	03/03/1978	Female	F2
388LMDV	Oscar	Ciao	01/01/1987	Male	F2

(i) Write a Data Definition Language (DDL) statement to create the EMPLOYEES database.

..... [1]

(ii) Write a DDL statement to define the table EMPLOYEE_DATA, and declare EmployeeID as the primary key.

.....
.....
.....
.....
.....

AS/A Level Computer Science Exam Question 4:

- (iii) Write a Data Manipulation Language (DML) statement to return the first name and last name of all female employees in the department named Finance.

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

[5]

AS/A Level Computer Science Exam Answer 4:

Question	Answer	Marks
3(a)(i)	<p>1 mark per bullet point</p> <ul style="list-style-type: none"><input type="checkbox"/> Stores all the information about the database // data about the data // metadata about the data<input type="checkbox"/> For example, fields, data types, validation, keys	2
3(a)(ii)	<p>1 mark per bullet point to max 2</p> <ul style="list-style-type: none"><input type="checkbox"/> Allows the user to enter criteria<input type="checkbox"/> Searches for data which meets the entered criteria<input type="checkbox"/> Organises the results to be displayed to the user	2
3(b)	<p>1 mark per bullet point to max 2</p> <ul style="list-style-type: none"><input type="checkbox"/> Primary key uniquely identifies each tuple // Each tuple in the table is unique<input type="checkbox"/> Primary key can be used as a foreign key in another table<input type="checkbox"/> to form a link/relationship between the tables <p>By example:</p> <ul style="list-style-type: none"><input type="checkbox"/> Identification of a primary key in a table<input type="checkbox"/> Describing <u>that</u> primary key in another table as a foreign key	2

AS/A Level Computer Science Exam Answer 4:

Question	Answer	Marks
3(c)	<p>1 mark for each correct link</p> <pre>graph TD; A[EMPLOYEE_DATA] --> B[DEPARTMENT_MANAGER]; B --> C[DEPARTMENT]</pre>	3
3(d)	<p>1 mark per bullet point to max 3</p> <ul style="list-style-type: none"><input type="checkbox"/> There are no repeating groups (1NF)<input type="checkbox"/> There are no partial dependencies (2NF)<input type="checkbox"/> There are no non-key dependencies // There are no transitive dependencies (3NF)	3

AS/A Level Computer Science Exam Answer 4:

3(e)(i)	1 mark for correct answer CREATE DATABASE EMPLOYEES;	1
3(e)(ii)	1 mark per bullet <ul style="list-style-type: none"><input type="checkbox"/> Create table EMPLOYEE_DATA with open and close brackets<input type="checkbox"/> EmployeeID as VarChar restricted to max 7, Gender as a VarChar restricted to max 6, DepartmentNumber as a VarChar restricted to max 2<input type="checkbox"/> FirstName and LastName as VarChar (any max lengths must be reasonable)<input type="checkbox"/> Date of birth as Date<input type="checkbox"/> Declaring EmployeeID as PK CREATE TABLE EMPLOYEE_DATA (EmployeeID VarChar(7), FirstName VarChar, LastName VarChar, DateOfBirth Date, Gender VarChar(6), DepartmentNumber VarChar(2), PRIMARY KEY (EmployeeID)) ;	5

AS/A Level Computer Science Exam Answer 4:

Question	Answer	Marks
3(e)(iii)	<p>1 mark per bullet</p> <ul style="list-style-type: none"><input type="checkbox"/> Select FirstName and LastName only<input type="checkbox"/> From both tables<input type="checkbox"/> Where DepartmentName = "Finance"<input type="checkbox"/> AND Gender = "Female"<input type="checkbox"/> Joining tables (either AND, or inner join) <pre>SELECT FirstName, LastName FROM EMPLOYEE_DATA, DEPARTMENT WHERE DepartmentName = "Finance" AND Gender = "Female" AND DEPARTMENT.DepartmentNumber = EMPLOYEE_DATA.DepartmentNumber;</pre>	5

AS/A Level Computer Science Exam Question 5:

- A social media website has a relational database, WEBDATA, that stores the site's information.
The database has three tables to store users' details, and details of the images and text that they post.

USER (UserName, FirstName, SecondName, DateOfBirth)

PHOTO (PhotoID, UserName, Comment, UploadDate)

TEXTPOST (PostID, UserName, DateOfPost, TheText)

- (a) (i) Explain how the relationship between the tables USER and PHOTO has been implemented.

.....
.....
.....
.....

[2]

AS/A Level Computer Science Exam Question 5:

- (ii) Draw the entity-relationship (E-R) diagram to show the relationships between the three tables.

[2]

AS/A Level Computer Science Exam Question 5:

- (b) A database administrator decides to enforce referential integrity.

Use an example from the database WEBDATA to explain what is meant by **referential integrity**.

.....
.....
.....
.....
.....
.....
.....

[3]

AS/A Level Computer Science Exam Question 5:

- (c) The database has been normalised to Third Normal Form (3NF).

Define the three stages of database normalisation.

1NF.....

.....

2NF.....

.....

3NF.....

[3]

AS/A Level Computer Science Exam Question 5:

(d) The following shows sample data from the **USER** table.

UserName	FirstName	SecondName	DateOfBirth
gem123	John	Smith	01/01/1995
purpleSky	Muhammed	Ali	23/02/1956
OpenWindow	Sunny	Amir	03/03/1997
bluebird127	Raziya	Bello	04/03/1982

(i) Write an SQL script to create the **USER** table.

.....

.....

.....

.....

.....

.....

.....

AS/A Level Computer Science Exam Question 5:

- (ii) The database administrator needs to alter the USER table. A new field, country, needs to be added.

Write an SQL script to add the field Country to the USER table.

```
.....  
.....  
.....  
.....
```

[2]

AS/A Level Computer Science Exam Answer 5:

Question	Answer	Marks
7(a)(i)	<p>1 mark per bullet</p> <ul style="list-style-type: none"><input type="checkbox"/> <u>UserName</u> is the <u>primary key</u> in <u>USER</u><input type="checkbox"/> <u>UserName</u> is (included as) a <u>foreign key</u> in <u>PHOTO</u>	2
7(a)(ii)	<p>1 mark for each correct relationship</p> <pre>graph LR; PHOTO --> USER; USER <--> TEXTPOST;</pre> The diagram illustrates the relationships between three entities: PHOTO, USER, and TEXTPOST. Entity boxes are labeled PHOTO, USER, and TEXTPOST. Relationships are shown by lines connecting the boxes. A line connects PHOTO to USER with an arrow pointing from PHOTO to USER, indicating a one-to-many relationship. Another line connects USER to TEXTPOST with an arrow pointing from USER to TEXTPOST, indicating another one-to-many relationship.	2

AS/A Level Computer Science Exam Answer 5:

Question	Answer	Marks
7(b)	<p>1 mark per bullet to max 2 for explanation</p> <ul style="list-style-type: none"><input type="checkbox"/> Referential integrity is making sure tables do not try to reference data which does not exist // A value of one attribute of a table exists as a value of another attribute in a different table<input type="checkbox"/> A primary key cannot be deleted unless all dependent records are already deleted<input type="checkbox"/> Cascading delete<input type="checkbox"/> A primary key cannot be updated unless all dependent records are already updated<input type="checkbox"/> Cascading update / edit<input type="checkbox"/> Every foreign key value has a matching value in the corresponding primary key<input type="checkbox"/> The foreign keys must be the same data type as the corresponding primary key <p>1 mark for a suitable example</p> <p>e.g.</p> <ul style="list-style-type: none"><input type="checkbox"/> A <code>UserName</code> cannot be deleted from the <code>USER</code> table if they have a related <code>photo/textpost</code><input type="checkbox"/> If <code>UserName</code> is updated in <code>USER</code> table, it must also be updated in <code>PHOTO</code> and <code>TEXTPOST</code> tables<input type="checkbox"/> Cannot create/edit a record in <code>TEXTPOST / PHOTO</code> without a matching entry in <code>USER</code> table	3

AS/A Level Computer Science Exam Answer 5:

7(c)	<p>Max 1 mark from each bulleted group</p> <p>1NF</p> <ul style="list-style-type: none"><input type="checkbox"/> No repeated groups of attributes<input type="checkbox"/> All attributes should be atomic<input type="checkbox"/> No duplicate rows <p>2NF (in 1NF and)</p> <ul style="list-style-type: none"><input type="checkbox"/> No partial dependencies <p>3NF (in 2NF and)</p> <ul style="list-style-type: none"><input type="checkbox"/> No non-key dependencies<input type="checkbox"/> No transitive dependencies	3
7(d)(i)	<p>1 mark per bullet</p> <ul style="list-style-type: none"><input type="checkbox"/> CREATE TABLE USER and () ;<input type="checkbox"/> UserName, FirstName and SecondName as VARCHAR and commas<input type="checkbox"/> DateOfBirth as DATE and comma<input type="checkbox"/> PRIMARY KEY(UserName)<input type="checkbox"/> An appropriate NOT NULL <pre>CREATE TABLE USER(UserName: varchar(15) NOT NULL, FirstName: varchar(25), SecondName: varchar(25), DateOfBirth: Date, PRIMARY KEY(UserName));</pre>	5

AS/A Level Computer Science Exam Answer 5:

Question	Answer	Marks
7(d)(ii)	<p>1 mark per bullet</p> <ul style="list-style-type: none"><input type="checkbox"/> ALTER TABLE USER<input type="checkbox"/> ADD COUNTRY varchar; <p>ALTER TABLE USER ADD Country varchar;</p>	2

AS/A Level Computer Science Exam Question 6:

A company writes applications (apps) for smartphones. The company has a relational database, PURPLEGAME, which stores the information for one of its online game apps.

The database has three tables to store player's details, dates when they have logged into the app and in-app purchase details.

LOGIN (LoginID, PlayerID, Date)

PURCHASE (PurchaseID, PlayerID, PurchaseDate, Cost)

PLAYER (PlayerID, PlayerName, SkillLevel)

- (a) Draw the entity-relationship (E-R) diagram to show the relationships between the three tables.

[2]

AS/A Level Computer Science Exam Question 6:

- (b) The database manager is concerned about data integrity.

State what is meant by **data integrity**. Give an example of how the manager can ensure data integrity in the PURPLEGAME database.

.....
.....
.....
.....

[2]

- (c) The database designer states that the PURPLEGAME database is in Third Normal Form (3NF).

Tick () **one** box to indicate whether this statement is true or false.

True	False

Justify your choice.

.....
.....
.....
.....
.....

[3]

AS/A Level Computer Science Exam Question 6:

(d) (i) The following table shows some sample data for the **PLAYER** table.

PlayerID	PlayerName	SkillLevel
fly918	Kylie	3
elephant11	Mehrdad	9
candy22	Suzi	15
greenGrass	Jason	22

Write an SQL script to create the **PLAYER** table.

.....
.....
.....
.....
.....
.....
.....
.....

AS/A Level Computer Science Exam Question 6:

- (ii) The table, PLAYER, needs to be altered. A new field, DateofBirth, needs to be added.

Write an SQL script to add the DateofBirth field to the PLAYER table.

```
.....  
.....  
.....  
.....
```

[2]

AS/A Level Computer Science Exam Answer 6:

Question	Answer	Marks
2(a)	<p>1 mark for each correct relationship</p> <pre>graph LR; LOGIN --> PLAYER; PLAYER --> PURCHASE; PURCHASE --> LOGIN;</pre> <p>The diagram shows three rectangular boxes representing classes: 'LOGIN', 'PLAYER', and 'PURCHASE'. Arrows indicate directed relationships: an arrow from 'LOGIN' to 'PLAYER', another from 'PLAYER' to 'PURCHASE', and a final arrow from 'PURCHASE' back to 'LOGIN'.</p>	2
2(b)	<p>1 mark for description</p> <ul style="list-style-type: none"><input type="checkbox"/> Ensure data is consistent / accurate // keep data consistent / accurate <p>1 mark for example from:</p> <p>e.g.</p> <ul style="list-style-type: none"><input type="checkbox"/> Validation rules<input type="checkbox"/> Referential integrity<input type="checkbox"/> Verification<input type="checkbox"/> Input masks<input type="checkbox"/> Setting data types<input type="checkbox"/> Removing redundant data<input type="checkbox"/> Backup data<input type="checkbox"/> Access controls<input type="checkbox"/> Audit trail	2

AS/A Level Computer Science Exam Answer 6:

2(c)	<p>1 mark for the correct box ticked</p> <table border="1"><tr><td>True</td><td>False</td></tr><tr><td>✓</td><td></td></tr></table> <p>1 mark per bullet for justification, max 2</p> <ul style="list-style-type: none"><input type="checkbox"/> No repeated attributes // data is atomic // No partial dependencies (no dual keys)<input type="checkbox"/> No non-key / transitive dependencies	True	False	✓		3
True	False					
✓						

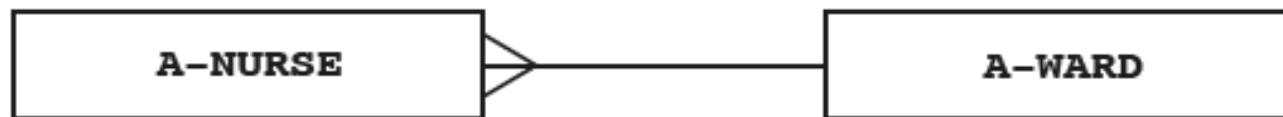
AS/A Level Computer Science Exam Answer 6:

Question	Answer	Marks
2(d)(i)	<p>1 mark per bullet</p> <ul style="list-style-type: none"><input type="checkbox"/> CREATE TABLE PLAYER and () ;<input type="checkbox"/> PlayerID and PlayerName as VARCHAR and commas<input type="checkbox"/> SkillLevel as INT and comma<input type="checkbox"/> PRIMARY KEY(PlayerID)<input type="checkbox"/> An appropriate NOT NULL <pre>CREATE TABLE PLAYER(PlayerID: varchar NOT NULL, PlayerName: varchar, SkillLevel: int, PRIMARY KEY(PlayerID),);</pre>	5
2(d)(ii)	<p>1 mark per bullet</p> <ul style="list-style-type: none"><input type="checkbox"/> ALTER TABLE PLAYER<input type="checkbox"/> ADD DateOfBirth Date;	2

AS/A Level Computer Science Exam Question 7:

A hospital is divided into two areas, Area A and Area B. Each area has several wards. All the ward names are different.

A number of nurses are based in Area A. These nurses always work on the same ward. Each nurse has a unique Nurse ID of STRING data type.



- (a) Describe the relationship shown above.

.....
.....

[1]

AS/A Level Computer Science Exam Question 7:

- (b) A relational database is created to store the ward and nurse data. The two table designs for Area A are:

A-WARD (WardName, NumberOfBeds)

A-NURSE (NurseID, FirstName, FamilyName,))

- (i) Complete the design for the A-NURSE table. [1]
- (ii) Explain how the relationship in part (a) is implemented.

.....

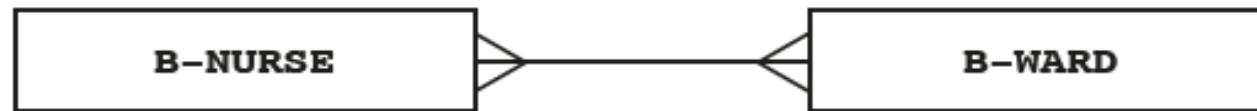
.....

.....

..... [2]

AS/A Level Computer Science Exam Question 7:

- (c) In Area B of the hospital, there are a number of wards and a number of nurses.
 - Each Area B ward has a specialism.
 - Each Area B nurse has a specialism.
 - A nurse can be asked to work in any of the Area B wards where their specialism matches with the ward specialism.
- The relationship for Area B of the hospital is:



- (i) Explain what the degree of relationship is between the entities B-NURSE and B-WARD.

.....
.....

[1]

AS/A Level Computer Science Exam Question 7:

- (ii) The design for the Area B data is as follows:

B-NURSE (NurseID, FirstName, FamilyName, Specialism)

B-WARD (WardName, NumberOfBeds, Specialism)

B-WARD-NURSE (.....)

Complete the attributes for the third table. Underline its primary key.

[2]

- (iii) Draw the relationships on the entity-relationship (E-R) diagram.



[2]

AS/A Level Computer Science Exam Question 7:

- (d) Use the table designs in part (c)(ii).

- (i) Write an SQL query to display the Nurse ID and family name for all Area B nurses with a specialism of 'THEATRE'.

.....
.....
.....

[3]

- (ii) Fatima Woo is an Area B nurse with the nurse ID of 076. She has recently married, and her new family name is Chi.

Write an SQL command to update her record.

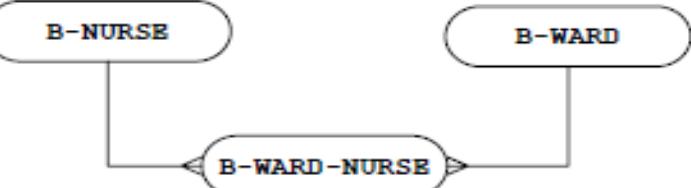
UPDATE

SET

WHERE

[3]

AS/A Level Computer Science Exam Answer 7:

Question	Answer	Marks
1(a)	Many-to-one	1
1(b)(i)	A-NURSE (<u>NurseID</u> , FirstName, FamilyName, WardName)	1
1(b)(ii)	<input type="checkbox"/> The primary key <u>WardName</u> in the A-WARD table ... <input type="checkbox"/> ... links to the foreign key <u>WardName</u> in the A-NURSE table.	1 1 2
1(c)(i)	Many-to-many relationship	1
1(c)(ii)	B-WARD-NURSE (<u>WardName</u> , <u>NurseID</u>) Both attributes (with no additions) Joint primary key correctly underlined	2 1 1
1(c)(iii)	 <pre> graph TD B_NURSE((B-NURSE)) <--> > B_WARD((B-WARD)) B_WARD --- B_WARD_NURSE((B-WARD-NURSE)) B_NURSE --- B_WARD_NURSE </pre> <p>Correct relationship between B-NURSE and B-WARD-NURSE Correct relationship between B-WARD and B-WARD-NURSE</p>	2
1(d)(i)	<pre> SELECT NurseID, FamilyName FROM B-NURSE WHERE Specialism = 'THEATRE'; </pre>	1 1 1 3
1(d)(ii)	<pre> UPDATE B-NURSE SET FamilyName = 'Chi' WHERE NurseID = '076'; </pre>	1 1 1 3

AS/A Level Computer Science Exam Question 8:

Some shops belong to the Rainbow Retail buying group. They buy their goods from one or more suppliers.

Each shop has:

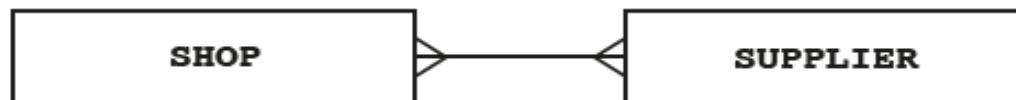
- a unique shop ID
- a single retail specialism (for example, food, electrical, garden).

Each supplier has:

- a unique supplier ID
- a similar single specialism recorded.

Rainbow Retail creates a relational database to record data about the shops and their suppliers.

The entity-relationship (E-R) diagram for the relationship between the **SHOP** and **SUPPLIER** tables is shown.



- (a) Explain what the degree of relationship is between the entities **SHOP** and **SUPPLIER**.

.....
.....

[1]

AS/A Level Computer Science Exam Question 8:

The database design is as follows:

SHOP (ShopID, ShopName, Location, RetailSpecialism)

SUPPLIER (SupplierID, SupplierName, ContactPerson, RetailSpecialism)

SHOP-SUPPLIER (ShopID, SupplierID)

The SHOP-SUPPLIER table stores the suppliers that each shop has previously used.

Primary keys are not shown.

- (b) (i) Label the entities and draw the relationships to complete the revised E-R diagram.



[3]

AS/A Level Computer Science Exam Question 8:

(ii) Complete the following table to show for each database table:

- the primary key
- the foreign key(s) (if any):
 - Each table may contain none, one or more foreign key(s).
 - For a table with no foreign key, write ‘None’.
- an explanation for the use of any foreign key.

Table	Primary key	Foreign key(s) (if any)	Explanation
SHOP			
SUPPLIER			
SHOP-SUPPLIER			

[5]

AS/A Level Computer Science Exam Question 8:

- (iii) The database designer has implemented SUPPLIER.ContactPerson as a secondary key.

Describe the reason for this.

[2]

- (c) (i) Write an SQL query to display the shop ID and location of all shops with a 'GROCERY' specialism.

[3]

- (ii) The existing shop with ID 8765 has just used the existing supplier SUP89 for the first time.

Write an SQL script to add this data to the database.

[3]

AS/A Level Computer Science Exam Answer 8:

Question	Answer	Marks
1(a)	Many-to-many relationship	1
1(b)(i)	<p>The diagram illustrates a many-to-many relationship between two entities: SHOP and SUPPLIER. Both entities are represented by rounded rectangles. A third rounded rectangle, labeled "SHOP-SUPPLIER", contains a horizontal double-headed arrow, indicating a bidirectional relationship between the two entities.</p> <p>Both entities correctly labelled Correct relationship between SHOP and SHOP-SUPPLIER Correct relationship between SUPPLIER and SHOP-SUPPLIER</p>	3
	1	
	1	
	1	

AS/A Level Computer Science Exam Answer 8:

1(b)(ii)	Table	Primary key	Foreign keys(s) (if any)	Explanation	5
SHOP	ShopID	None			
SUPPLIER	SupplierID	None			
SHOP-SUPPLIER	ShopID AND SupplierID	ShopID OR SupplierID (or both)	To create a link with the SHOP or SUPPLIER table.		

<input type="checkbox"/> SHOP has primary key ShopID and SUPPLIER has primary key SupplierID	1
<input type="checkbox"/> SHOP-SUPPLIER has primary key ShopID + SupplierID	1
<input type="checkbox"/> Both SHOP and SUPPLIER show foreign key as 'None'	1
<input type="checkbox"/> SHOP-SUPPLIER shows foreign key ShopID or SupplierID	1
<input type="checkbox"/> Explanation for SHOP-SUPPLIER foreign key describes ShopID or SupplierID creating a link	1

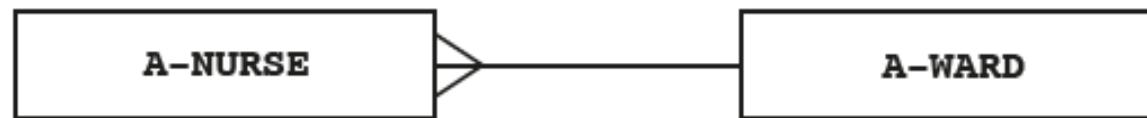
AS/A Level Computer Science Exam Answer 8:

1(b)(iii)	Two from: <input type="checkbox"/> The database user will <u>frequently</u> want to search on contact name <input type="checkbox"/> The contact name attribute has been indexed <input type="checkbox"/> It allows for a <u>fast/faster</u> search using contact name	1	1	1	Max 2
1(c)(i)	SELECT ShopID, Location FROM SHOP WHERE RetailSpecialism = 'GROCERY';	1	1	1	3
1(c)(ii)	INSERT INTO SHOP-SUPPLIER (ShopID, SupplierID) VALUES (8765, 'SUP89');	1	1	1	3

AS/A Level Computer Science Exam Question 9:

- A hospital is divided into two areas, Area A and Area B. Each area has several wards. All the ward names are different.

A number of nurses are based in Area A. These nurses always work on the same ward. Each nurse has a unique Nurse ID of STRING data type.



- (a)** Describe the relationship shown above.

.....
.....

[1]

AS/A Level Computer Science Exam Question 9:

- . (b) A relational database is created to store the ward and nurse data. The two table designs for Area A are:

A-WARD (WardName, NumberOfBeds)

A-NURSE (NurseID, FirstName, FamilyName,)

- (i) Complete the design for the A-NURSE table. [1]
- (ii) Explain how the relationship in part (a) is implemented.

.....

.....

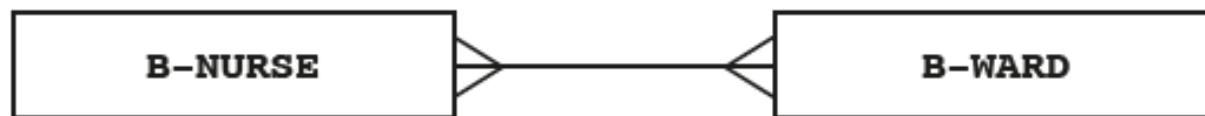
.....

..... [2]

AS/A Level Computer Science Exam Question 9:

- (c) In Area B of the hospital, there are a number of wards and a number of nurses.
 - Each Area B ward has a specialism.
 - Each Area B nurse has a specialism.
 - A nurse can be asked to work in any of the Area B wards where their specialism matches with the ward specialism.

The relationship for Area B of the hospital is:



- (i) Explain what the degree of relationship is between the entities B-NURSE and B-WARD.

.....
.....

[1]

AS/A Level Computer Science Exam Question 9:

- (ii) The design for the Area B data is as follows:

B-NURSE (NurseID, FirstName, FamilyName, Specialism)

B-WARD (WardName, NumberOfBeds, Specialism)

B-WARD-NURSE (.....)

Complete the attributes for the third table. Underline its primary key.

[2]

- (iii) Draw the relationships on the entity-relationship (E-R) diagram.

B-NURSE

B-WARD

B-WARD-NURSE

[2]

AS/A Level Computer Science Exam Question 9:

- (d) Use the table designs in part (c)(ii).

- (i) Write an SQL query to display the Nurse ID and family name for all Area B nurses with a specialism of 'THEATRE'.

.....
.....
.....

[3]

- (ii) Fatima Woo is an Area B nurse with the nurse ID of 076. She has recently married, and her new family name is Chi.

Write an SQL command to update her record.

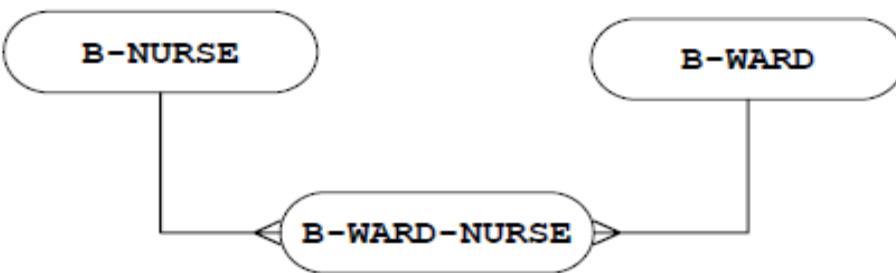
UPDATE

SET

WHERE

[3]

AS/A Level Computer Science Exam Answer 9:

Question	Answer	Marks
1(a)	Many-to-one	1
1(b)(i)	A-NURSE (<u>NurseID</u> , FirstName, FamilyName, WardName)	1
1(b)(ii)	<input type="checkbox"/> The primary key <u>WardName</u> in the A-WARD table ... <input type="checkbox"/> ... links to the foreign key <u>WardName</u> in the A-NURSE table.	1 1 2
1(c)(i)	Many-to-many relationship	1
1(c)(ii)	B-WARD-NURSE (<u>WardName</u> , <u>NurseID</u>) Both attributes (with no additions) Joint primary key correctly underlined	1 1 2
1(c)(iii)	 Correct relationship between B-NURSE and B-WARD-NURSE Correct relationship between B-WARD and B-WARD-NURSE	1 1 2

AS/A Level Computer Science Exam Answer 9:

1(d)(i)	SELECT NurseID, FamilyName FROM B-NURSE WHERE Specialism = 'THEATRE';	1 1 1	3
1(d)(ii)	UPDATE B-NURSE SET FamilyName = 'Chi' WHERE NurseID = '076';	1 1 1	3

AS/A Level Computer Science Exam Question 10:

A Local Area Network is used by staff in a hospital to access data stored in a Database Management System (DBMS).

(a) Name **two** security measures to protect computer systems.

1

2

[2]

AS/A Level Computer Science Exam Question 10:

(b) A frequent task for staff is to key in new patient data from a paper document. The document includes the patient's personal ID number.

(i) The Patient ID is a seven digit number. The database designer decides to use a check digit to verify each foreign key value that a user keys in for a Patient ID.

When a user assigns a primary key value to a Patient ID, the DBMS adds a modulus-11 check digit as an eighth digit. The DBMS uses the weightings 6, 5, 4, 3, 2 and 1 for calculating the check digit. It uses 6 as the multiplier for the most significant (leftmost) digit.

Show the calculation of the check digit for the Patient ID with the first six digits 786531.

Complete Patient ID [4]

AS/A Level Computer Science Exam Question 10:

- (ii) Name and describe **two** validation checks that the DBMS could carry out on each primary key value that a user keys in for a Patient ID.

1 Validation check

Description

.....

2 Validation check

Description

.....

[4]

AS/A Level Computer Science Exam Answer 10:

Question	Answer	Marks																																													
3(a)	<p>Two marks from:</p> <ul style="list-style-type: none"><input type="checkbox"/> Physical measures<input type="checkbox"/> Access rights<input type="checkbox"/> Encryption<input type="checkbox"/> Firewall<input type="checkbox"/> Use authentication methods such as usernames and passwords<input type="checkbox"/> Anti-malware program	Max 2																																													
3(b)(i)	<table border="1"><tbody><tr><td>7</td><td>X</td><td>6</td><td>=</td><td>42</td></tr><tr><td>8</td><td>X</td><td>5</td><td>=</td><td>40</td></tr><tr><td>6</td><td>X</td><td>4</td><td>=</td><td>24</td></tr><tr><td>5</td><td>X</td><td>3</td><td>=</td><td>15</td></tr><tr><td>3</td><td>X</td><td>2</td><td>=</td><td>6</td></tr><tr><td>1</td><td>X</td><td>1</td><td>=</td><td>1</td></tr><tr><td></td><td></td><td></td><td>Total:</td><td>128 / 11</td></tr><tr><td></td><td></td><td></td><td></td><td>11 R 7</td></tr><tr><td></td><td></td><td></td><td>Check digit:</td><td>11 - 7 = 4</td></tr></tbody></table> <p>1 mark for 6 values 1 mark for 2 steps Accept $128 \text{ MOD } 11 = 7$ 1 mark for subtraction</p> <p>Answer: 786531 4 (1 mark for answer)</p>	7	X	6	=	42	8	X	5	=	40	6	X	4	=	24	5	X	3	=	15	3	X	2	=	6	1	X	1	=	1				Total:	128 / 11					11 R 7				Check digit:	11 - 7 = 4	4
7	X	6	=	42																																											
8	X	5	=	40																																											
6	X	4	=	24																																											
5	X	3	=	15																																											
3	X	2	=	6																																											
1	X	1	=	1																																											
			Total:	128 / 11																																											
				11 R 7																																											
			Check digit:	11 - 7 = 4																																											

AS/A Level Computer Science Exam Answer 10:

3(b)(ii)	<p>One mark for name of check One mark for description Max two checks</p> <p>Uniqueness check Each PatientID must be unique</p> <p>Length check Each PatientID is exactly 7 characters</p> <p>Format check / Type check All 7 characters must be <u>digits</u></p> <p>Presence check PatientID must be entered</p>	Max 4
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------

AS/A Level Computer Science Exam Question 11:

- A Local Area Network is used by school staff who access data stored in a Database Management System (DBMS).

(a) (i) Explain the difference between security and privacy of data.

.....
.....
.....
.....
.....

[3]

(ii) Give an example for this application where privacy of data is a key concern.

.....
.....

[1]

AS/A Level Computer Science Exam Question 11:

- (b) Name and describe **two** security measures the Network Manager has in place to protect the security of the data held in the DBMS.

1

.....

.....

.....

2

.....

.....

.....

[4]

AS/A Level Computer Science Exam Question 11:

(c) A task for staff at the start of the school year is to key in new pupil data from a paper document.

The data is entered to a screen form and includes the data verification of some fields.

Describe what is meant by **verification**.

.....
.....
.....
.....

[2]

AS/A Level Computer Science Exam Answer 11:

Question	Answer	Marks
3(a)(i)	<p>1 Mark per bullet, max 3</p> <ul style="list-style-type: none"><input type="checkbox"/> Security is keeping the data safe<input type="checkbox"/> From accidental / malicious damage /loss<input type="checkbox"/> By example of need for security <input type="checkbox"/> Privacy is the need to restrict access to personal data<input type="checkbox"/> To avoid it being seen by unauthorised people<input type="checkbox"/> By example of need for privacy	3
3(a)(ii)	<p>1 Mark for a suitable example</p> <p>For example: Personal data of students / staff</p>	1

AS/A Level Computer Science Exam Answer 11:

3(b)	<p>1 Mark for stating the security measure 1 Mark for a corresponding description Maximum 2 marks for each measure Maximum 2 measures</p> <p>Physical measures <input type="checkbox"/> Locked doors/keyboards etc. <input type="checkbox"/> Secure methods of access, keypads/ biometric scans etc.</p> <p>Backup of data <input type="checkbox"/> Regular copies of the data are made <input type="checkbox"/> If the data is corrupted it can be restored</p> <p>Disk-mirroring <input type="checkbox"/> All activity is duplicated to a second disk in real time so that if the first disk fails there is a complete copy available</p> <p>Access rights <input type="checkbox"/> Different access rights for individuals/groups of users <input type="checkbox"/> To stop users editing data they are not permitted to access <input type="checkbox"/> By example</p> <p>Encryption <input type="checkbox"/> If accessed, data cannot be understood by unauthorised personnel <input type="checkbox"/> Accessed only by those with the decryption key</p> <p>Firewall <input type="checkbox"/> To stop unauthorised access/hackers gaining access to the computer network</p> <p>Use authentication methods such as passwords and usernames <input type="checkbox"/> Passwords should be strong / biometrics <input type="checkbox"/> To prevent unauthorised access to data</p> <p>Anti-malware program <input type="checkbox"/> To detect / remove / quarantine viruses / key-loggers etc. <input type="checkbox"/> Carrying out regular scans</p> <p>Concurrent Access Controls // Record locking <input type="checkbox"/> Closes a record to second user until first update complete <input type="checkbox"/> To prevent simultaneous updates being lost</p>	4
------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

AS/A Level Computer Science Exam Answer 11:

Question	Answer	Marks
3(c)	<p>1 Mark per bullet, max 2</p> <ul style="list-style-type: none"><input type="checkbox"/> Checking that the data entered matches / is consistent with that of the source.<input type="checkbox"/> Comparison of two versions of the data<input type="checkbox"/> Examples include double entry, visual checking, proof reading etc...<input type="checkbox"/> In the event of a mismatch – the user is forced to re-enter the data<input type="checkbox"/> By example, e.g. creation of a password<input type="checkbox"/> Does not check data is sensible/acceptable	2

AS/A Level Computer Science Exam Question 12:

A school stores a large amount of data. This includes student attendance, qualification, and contact details. The school's software uses a file-based approach to store this data.

(a) The school is considering changing to a DBMS.

(i) State what DBMS stands for.

..... [1]

(ii) Describe **two** ways in which the Database Administrator (DBA) could use the DBMS software to ensure the security of the student data.

1

.....

.....

2

.....

.....

..... [4]

AS/A Level Computer Science Exam Question 12:

- (iii) A feature of the DBMS software is a query processor.

Describe how the school secretary could use this software.

.....
.....
.....
.....

[2]

- (iv) The DBMS has replaced software that used a file-based approach with a relational database.

Describe how using a relational database has overcome the previous problems associated with a file-based approach.

.....
.....
.....
.....

[3]

AS/A Level Computer Science Exam Question 12:

- (b) The database design has three tables to store the classes that students attend.

STUDENT (StudentID, FirstName, LastName, Year, TutorGroup)

CLASS (ClassID, Subject)

CLASS-GROUP (StudentID, ClassID)

Primary keys are not shown.

There is a one-to-many relationship between **CLASS** and **CLASS-GROUP**.

- (i) Describe how this relationship is implemented.

.....
.....
.....

[2]

- (ii) Describe the relationship between **CLASS-GROUP** and **STUDENT**.

.....

[1]

AS/A Level Computer Science Exam Question 12:

- (iii) Write an SQL script to display the `StudentID` and `FirstName` of all students who are in the tutor group 10B. Display the list in alphabetical order of `LastName`.

.....
.....
.....
.....

[4]

- (iv) Write an SQL script to display the `LastName` of all students who attend the class whose `ClassID` is CS1.

.....
.....
.....
.....
.....
.....
.....
.....

[4]

AS/A Level Computer Science Exam Answer 12:

(a) (i) Database Management System

[1]

(ii) One mark for identifying the way in which the data security is ensured, and one mark for a further description.

Maximum of two marks per method. Maximum of two methods.

[4]

- Issue usernames and passwords...
 - stops unauthorised access to the data
 - any further expansion e.g. strong passwords / passwords should be changed regularly etc...
- Access rights / privileges...
 - so that only relevant staff / certain usernames can read/edit certain parts of the data
 - can be read only, or full access / read, write and delete
 - any relevant example e.g. only class tutors can edit details of pupils in their tutor group

AS/A Level Computer Science Exam Answer 12:

- Create (regular / scheduled) backups...
 - in case of loss/damage to the live data a copy is available
 - any relevant example e.g. backing up the attendance registers at the end of each day and storing the data off-site/to a separate device
- Encryption of data...
 - if there is unauthorised access to the data it cannot be understood // needs a decryption key
 - any relevant example e.g. personal details of pupils are encrypted before being sent over the Internet to examination boards
- Definition of different views...
 - composed of one or more tables
 - controls the scope of the data accessible to authorised users
 - any relevant example e.g. teachers can only see their classes
- Usage monitoring / logging of activity...
 - creation of an audit /activity log
 - records the use of the data in the database / records operations performed by all users / all access to the data
 - any relevant example, e.g. Track who changed a student's grade

AS/A Level Computer Science Exam Answer 12:

(iii) Two points from:

[2]

- Set up search criteria
- To find / retrieve / return the data that matches the criteria
- Any relevant example e.g. find pupils who were absent on a particular day

(iv) Three points from:

[3]

- By storing data in (separate) linked tables data redundancy is reduced / data duplication is controlled...
- Compatibility / data integrity issues are reduced as data only needs to be updated once / is only stored once.
- Unwanted or accidental deletion of linked data is prevented as the DBMS will flag an error.
- Program - data dependence is overcome.
- Changes made to the structure of the data have little effect on existing programs.
- Ad-hoc / complex queries can be more easily made as the DBMS will have a query language/ QBE form.
- Unproductive maintenance is eliminated as changes only need to be made once (rather than changing multiple programs).
- Fields can be added or removed without any effect on existing programs (that do not use these fields).
- Security / privacy of the data is improved as each application only has access to the fields it needs.
- There is better control of data integrity as the DBMS (uses its Data Dictionary) to perform validation checks on data entered.

AS/A Level Computer Science Exam Answer 12:

(b) (i) Two points from: [2]

- The Primary Key in CLASS is ClassID
- The Foreign Key of CLASS-GROUP is ClassID.
- The Primary Key of CLASS is also included in CLASS-GROUP as a Foreign Key, (which links to CLASS table)

(ii) Many-to-one [1]

(iii) One mark per statement. Several statements may be on the same line. [4]

```
SELECT StudentID, FirstName  
FROM STUDENT  
WHERE TutorGroup = "10B" // WHERE (TutorGroup = "10B")  
ORDER BY LastName ASC;
```

AS/A Level Computer Science Exam Answer 12:

(iv) One mark per statement. Several statements may be on the same line.

[4]

```
SELECT STUDENT.LastName  
FROM STUDENT, CLASS-GROUP  
WHERE ClassID = "CS1" // WHERE (ClassID = "CS1")  
AND CLASS-GROUP.StudentID = STUDENT.StudentID;
```

One mark per statement. Several statements may be on the same line.

```
SELECT STUDENT.LastName  
FROM STUDENT INNER JOIN CLASS-GROUP  
ON CLASS-GROUP.StudentID = STUDENT.StudentID  
WHERE ClassID = "CS1" // WHERE (ClassID = "CS1");
```

AS/A Level Computer Science Exam Question 13:

- (a) A Database Management System (DBMS) provides the following features.

Draw a line to match each feature with its description.

Feature	Description
Data dictionary	A file or table containing all the details of the database design
Data security	Data design features to ensure the validity of data in the database
Data integrity	A model of what the database will look like, although it may not be stored in this way
	Methods of protecting the data including the uses of passwords and different access rights for different users of the database

[3]

AS/A Level Computer Science Exam Question 13:

A school stores a large amount of data that includes student attendance, qualification and contact details. The school is setting up a relational database to store these data.

- (b) The school needs to safeguard against any data loss.

Describe **three** factors to consider when planning a backup procedure for the data.

Justify your decisions.

1

.....

.....

2

.....

.....

3

.....

.....

[6]

AS/A Level Computer Science Exam Question 13:

- (c) The database design has three tables to store the qualifications and grades each student has attained. The following is a sample of the data from each table.

STUDENT

StudentID	FirstName	LastName	Tutor
001AT	Ahmad	Tan	11A
003JL	Jane	Li	11B
011HJ	Heather	Jones	10A

QUALIFICATION

QualCode	Level	Subject
CS1	IGCSE	Computer Science
MT9	IGCSE	Maths
SC12	IGCSE	Science

AS/A Level Computer Science Exam Question 13:

STUDENT-QUALIFICATION

QualCode	StudentID	Grade	DateOfAward
SC12	011HJ	A	31/8/2014
SC12	003JL	C	31/8/2014
CS1	003JL	B	31/8/2014

- (i) Draw an Entity-Relationship (E-R) diagram to show the relationships between these three tables.

[2]

- (ii) State the type of relationship that exists between STUDENT and STUDENT-QUALIFICATION.

.....[1]

AS/A Level Computer Science Exam Question 13:

- . (ii) Write an SQL script to display the StudentID, Grade and DateofAward for the QualCode value of SC12.

.....
.....
.....
.....
.....
.....

[3]

- (iii) Write an SQL script to display the FirstName and LastName and QualCode for all STUDENT-QUALIFICATIONS for which the Grade value is A.

.....
.....
.....
.....
.....
.....

AS/A Level Computer Science Exam Question 13:

- (iii) Describe how the relationship between QUALIFICATION and STUDENT-QUALIFICATION is implemented.**

.....
.....
.....
.....

[2]

- (d) (i) The database will store each student's date of birth.**

Write an SQL script to add a date of birth attribute to the appropriate table.

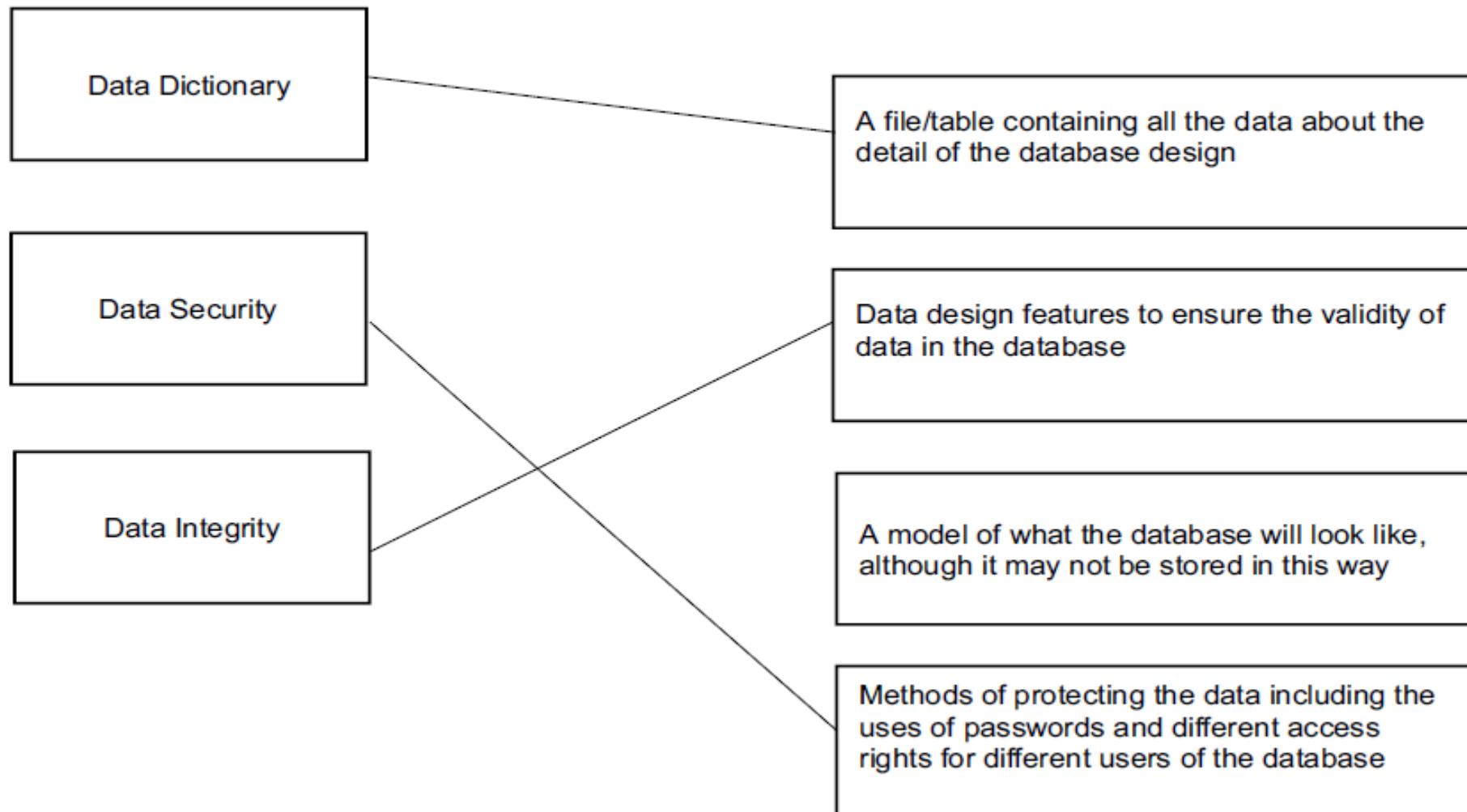
.....
.....
.....
.....

[2]

AS/A Level Computer Science Exam Answer 13:

5 (a) One mark for each correct line.

[3]



AS/A Level Computer Science Exam Answer 13:

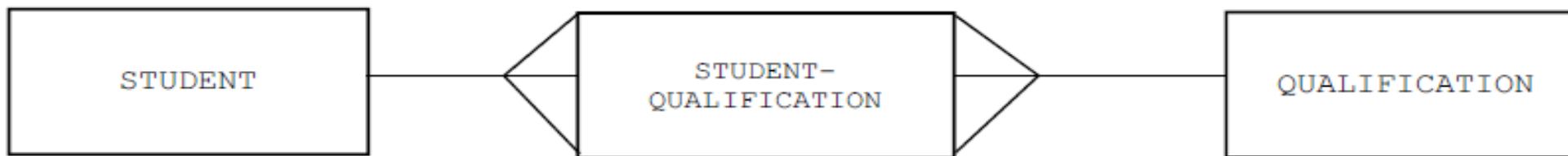
. (b) One mark for procedure point, one mark for justification. [6]

Maximum three procedures.

- How often should the data be backed up? e.g. at the end of each day
- Justification e.g. student's progress may be edited each day and should not be lost
- What medium should the data be backed up to? e.g. external hard disk drive
- Justification e.g. it has large enough capacity
- Where should the backups be stored? e.g. off-site
- Justification e.g. so if the building is damaged only the original data are lost
- What is backed up? e.g. only updated files ...
- Justification e.g. There are a large number of files and they are not all updated each day
- When should the backup take place? e.g. overnight
- Justification e.g. the system is not likely to be used then
- Who is responsible for performing the backup?
- Justification e.g. otherwise it may not be done
- Make sure the procedure is written down and understood by staff
- Justification e.g. otherwise some data may not be backed up

AS/A Level Computer Science Exam Answer 13:

(c) (i) One mark for each correct relationship. [2]



(ii) One-to-many [1]

(iii) Two points from: [2]

- The primary key in the QUALIFICATION table is QualCode.
- The foreign key in the STUDENT-QUALIFICATION table is QualCode.
- The primary key of QUALIFICATION is also included in QualCode.

(d) (i) One mark per statement. Several statements may be on one line. [2]

```
ALTER TABLE STUDENT  
ADD DateOfBirth DATE;
```

AS/A Level Computer Science Exam Answer 13:

- (ii) **One mark** per statement. Several statements may be on one line. [3]

```
SELECT StudentID, Grade, DateOfAward  
FROM STUDENT-QUALIFICATION  
WHERE QualCode = 'SC12';
```

- (iii) **One mark** per statement. Several statements may be on one line. [4]

```
SELECT STUDENT.FirstName, STUDENT.LastName, STUDENT-  
    QUALIFICATION.QualCode  
FROM STUDENT, STUDENT-QUALIFICATION  
WHERE STUDENT-QUALIFICATION.Grade = 'A'  
AND STUDENT.StudentID = STUDENT-QUALIFICATION.StudentID;
```

Alternative answer:

```
SELECT FirstName, LastName, STUDENT-QUALIFICATION.QualCode  
FROM STUDENT, INNER JOIN STUDENT-QUALIFICATION  
ON STUDENT.StudentID = STUDENT-QUALIFICATION.StudentID  
WHERE Grade = 'A';
```

AS/A Level Computer Science Exam Question 14:

(a) Five descriptions and seven relational database terms are shown below.

Draw a line to link each description to its correct database term.

Description	Database term
Any object, person or thing about which it is possible to store data	Secondary key
Dataset organised in rows and columns; the columns form the structure and the rows form the content	Candidate key
Any attribute or combination of attributes that can act as a unique key	Entity
Attribute(s) in a table that link to the primary key in another table to form a relationship	Foreign key
Attribute or combination of attributes that is used to uniquely identify a record	Primary key
	Table
	Tuple

[5]

AS/A Level Computer Science Exam Question 14:

- (b)** Explain what is meant by referential integrity.

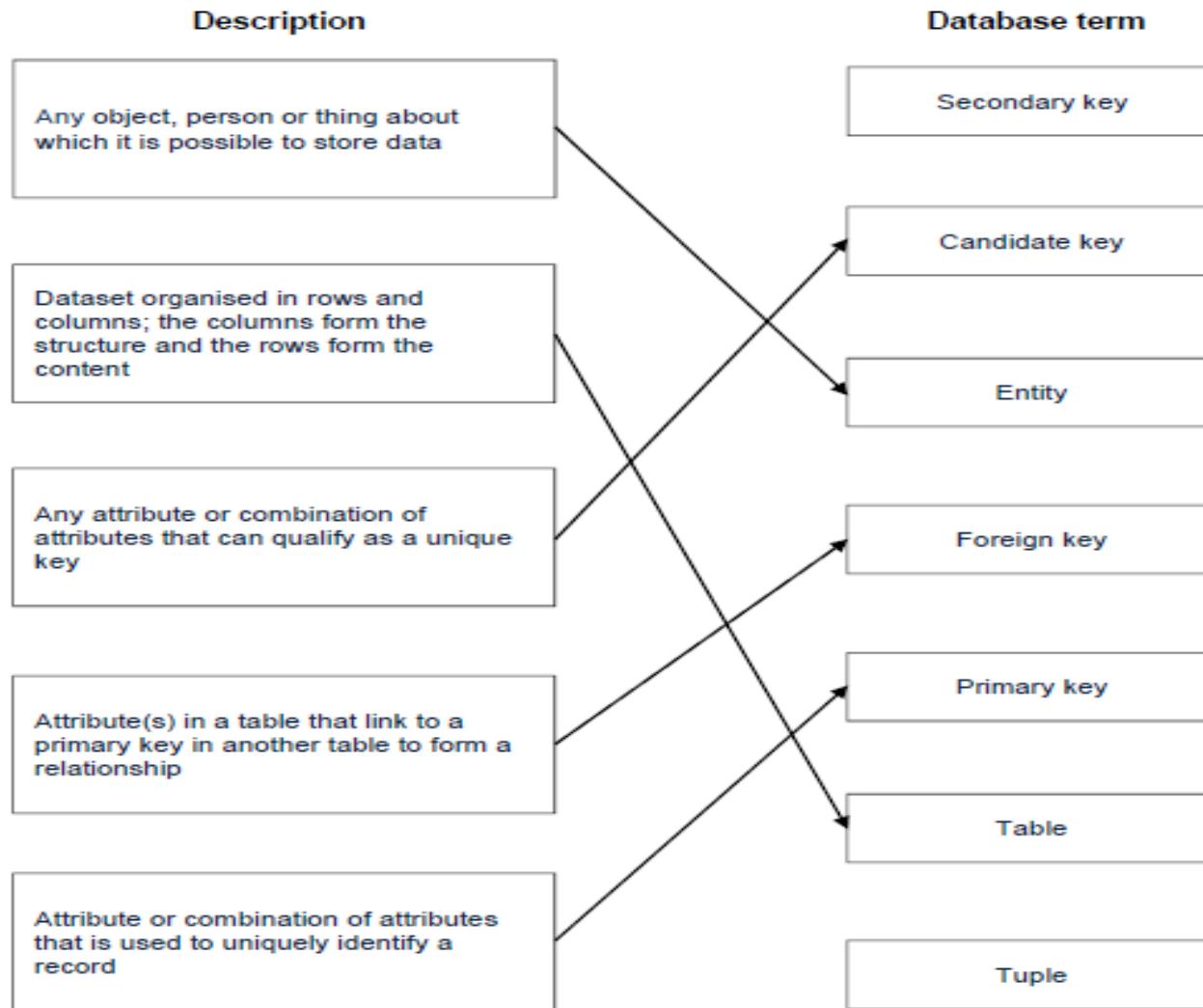
.....
.....
.....
.....
.....

[3]

AS/A Level Computer Science Exam Answer 14:

(a) One mark for each correct line.

Two lines from any box on left means no mark for that description.



[5]

AS/A Level Computer Science Exam Answer 14:

(b) Any **three** from:

- Ensures related data in tables are consistent
- If one table has a foreign key (the 'foreign' table)...
 - ... then it is not possible to add a record to that table / the 'foreign' table
 - ... unless there is a corresponding record in the linked table with a corresponding primary key (the 'primary' table)
- Cascading delete
- If a record is deleted in the 'primary' table...
 - all corresponding linked records in 'foreign' tables must also be deleted
- Cascading update
- If a record in the 'primary' table is modified...
 - ... all linked records in foreign tables will also be modified

[3]

AS/A Level Computer Science Exam Question 15:

A health club offers classes to its members. A member needs to book into each class in advance.

- (a) The health club employs a programmer to update the class booking system. The programmer has to decide how to store the records. The choice is between using a relational database or a file-based approach.

Give **three** reasons why the programmer should use a relational database.

1

.....

.....

2

.....

.....

3

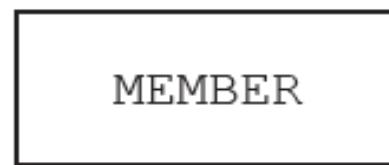
.....

.....

AS/A Level Computer Science Exam Question 15:

(b) The programmer decides to use three tables: MEMBER, BOOKING and CLASS.

Complete the Entity-Relationship (E-R) diagram to show the relationships between these tables.



[2]

AS/A Level Computer Science Exam Question 15:

(c) The CLASS table has primary key ClassID and stores the following data:

ClassID	Description	StartDate	ClassTime	NoOfSessions	AdultsOnly
DAY01	Yoga beginners	12/01/2016	11:00	5	TRUE
EVE02	Yoga beginners	12/01/2016	19:00	5	FALSE
DAY16	Circuits	30/06/2016	10:30	4	FALSE

Write an SQL script to create the CLASS table.

.....

.....

.....

.....

.....

.....

.....

AS/A Level Computer Science Exam Answer 15:

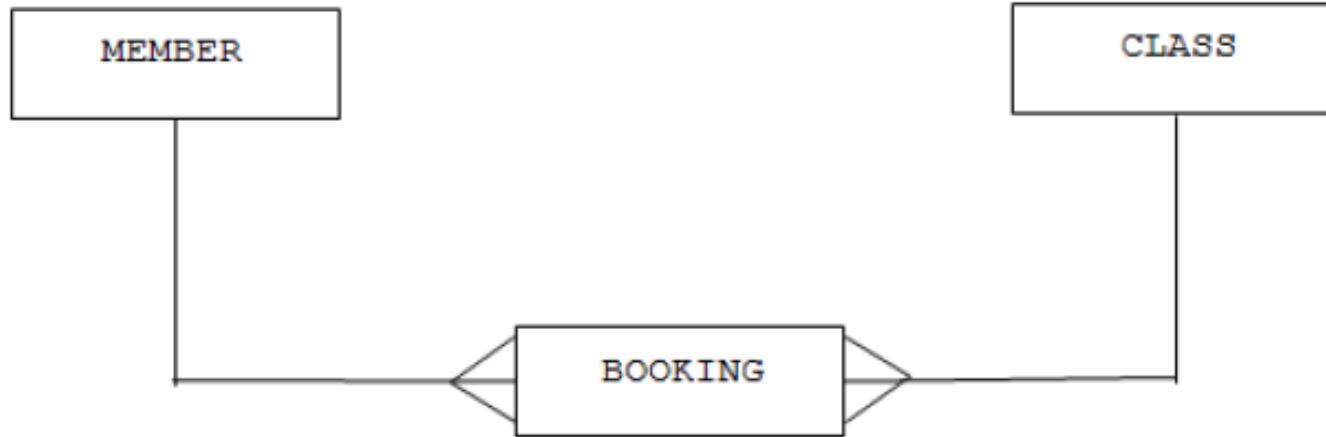
(a) ONE mark for each reason and ONE mark for a further explanation. MAX THREE reasons.

- Reduced data redundancy / data duplication
- Data is stored in (separate) linked tables
- The database (generally) stores data only once / data need only be updated once
- Improved data consistency/integrity/associated data will be automatically updated/easier to maintain the data/elimination of unproductive maintenance
- Complex queries can be more easily written
- To search/find specific data // specific example related to the Health Club
- Fields can be more easily added to or removed from tables
- Without affecting existing applications (that do not use these fields)
- Program-data dependence is overcome
- Changes to the data (design) do not require changes to programs // changes to programs do not require changes to data // the data can be accessed by any appropriate program
- Security is improved
- Each application only has access to the fields it needs // different users can be given different access rights
- Different users can be given different views of the data / data privacy is maintained
- So they do not see confidential information
- Allows concurrent access
- Record locking prevents two users updating the same record at the same time // record locking assures data consistency

[6]

AS/A Level Computer Science Exam Answer 15:

- (b) **ONE** mark for each correct relationship as shown.



[2]

AS/A Level Computer Science Exam Answer 15:

- (c) An example of a script is shown, but different syntax may be used.

```
CREATE TABLE CLASS (
    ClassID VARCHAR(5),
    Description VARCHAR(30),
    StartDate DATE,
    ClassTime TIME,
    NoOfSessions INT,
    AdultsOnly BIT,
    PRIMARY KEY(ClassID)
);
```

Mark as follows:

1 mark for CREATE TABLE CLASS **and () ;**
1 mark for PRIMARY KEY(ClassID)
1 mark for both ClassID VARCHAR(5), **and** Description VARCHAR(30),
1 mark for both StartDate DATE, **and** ClassTime TIME,
1 mark for NoOfSessions INT,
1 mark for AdultsOnly BIT,

[6]

AS/A Level Computer Science Exam Question 16:

A database has been designed to store data about salespersons and the products they have sold.

The following facts help to define the structure of the database:

- each salesperson works in a particular shop
- each salesperson has a unique first name
- each shop has one or more salespersons
- each product which is sold is manufactured by one company only
- each salesperson can sell any of the products
- the number of products that each salesperson has sold is recorded

The table `ShopSales` was the first attempt at designing the database.

FirstName	Shop	ProductName	NoOfProducts	Manufacturer
Nick	TX	television set refrigerator digital camera	3 2 6	SKC WP HKC
Sean	BH	hair dryer electric shaver	1 8	WG BG
John	TX	television set mobile phone digital camera toaster	2 8 4 3	SKC ARC HKC GK

(a) State why the table is **not** in First Normal Form (1NF).

.....
.....

[1]

AS/A Level Computer Science Exam Question 16:

- (b) The database design is changed to:

SalesPerson (FirstName, Shop)

SalesProducts (FirstName, ProductName, NoOfProducts, Manufacturer)

Using the data given in the first attempt table (ShopSales), show how these data are now stored in the revised table designs.

Table: SalesPerson

FirstName	Shop

AS/A Level Computer Science Exam Question 16:

Table: SalesProducts

FirstName	ProductName	NoOfProducts	Manufacturer

[3]

AS/A Level Computer Science Exam Question 16:

- (c) (i)** A relationship between the two tables has been implemented.

Explain how this has been done.

.....
.....
.....
.....
.....

[2]

- (ii)** Explain why the SalesProducts table is **not** in Third Normal Form (3NF).

.....
.....
.....
.....

[2]

- (iii)** Write the table definitions to give the database in 3NF.

.....
.....
.....

AS/A Level Computer Science Exam Answer 16:

(a) Any one from:

- (ShopSales) table has repeated group (of attributes)
- each sales person has a number of products
- FirstName, Shop would need to be repeated for each record

[1]

(b) One mark for SalesPerson table

table: SalesPerson

FirstName	Shop
Nick	TX
Sean	BH
John	TX

AS/A Level Computer Science Exam Answer 16:

table: SalesProducts

FirstName	ProductName	NoOfProducts	Manufacturer
Nick	television set	3	SKC
Nick	refrigerator	2	WP
Nick	digital camera	6	HKC
Sean	hair dryer	1	WG
Sean	electric shaver	8	BG
John	television set	2	SKC
John	mobile phone	8	ARC
John	digital camera	4	HKC
John	toaster	3	GK

(1 mark for FirstName column + 1 mark for remainder of table)

[3]

AS/A Level Computer Science Exam Answer 16:

(c) (i) Any two from:

- primary key of SalesPerson table is FirstName
- links to FirstName in SalesProducts table
- FirstName in SalesProductsS table is foreign key

[2]

(ii) • There is a non-key dependency
• Manufacturer is dependent on ProductName, (which is not the primary key of the SalesProducts table)

[2]

(iii) SalesPerson (FirstName, Shop)
-SalesProducts (FirstName, ProductName, NoOfProducts) OR
SalesProducts (SalesID, FirstName, ProductName, NoOfProducts)

-Product (ProductName, Manufacturer)

1 mark for correct attributes in SalesProducts and Product tables and 1 mark for correct identification of both primary keys

[2]

Glossary

application	A software program that allows a user to perform a specific task.
attribute	The characteristics of an entity. In databases, attributes are represented in fields, eg attributes of a film could be the actors, director and duration.
CSV	Comma-separated values - a standard file format for a flat-file database used in spreadsheet and database software.
data	Units of information. In computing there can be different data types, including integers, characters and Boolean. Data is often acted on by instructions.
database	A data store designed in an organised way, making it easier to search for the information you need.
element	A unique piece of data within a database.
encryption	Files that are encrypted have been altered using a secret code and are unreadable to unauthorised parties.
SQL	Structured query language - a programming language used to control databases.

Glossary

entity	An object, eg a person or film. In databases, entities are the subjects whose attributes are stored as records.
field	An element of a database record in which one piece of information is stored. For example 'name' in an electronic address book.
flat-file database	A database consisting of only one table often stored as a CSV file.
hard disk drive	A device used to store large amounts of data.
integrity	Refers to the validity of a database. The integrity can be damaged by changes to the structure or software bugs.
memory	The part of a computer that stores data.
MySQL	Open source database management system.
NoSQL	A form of non-relational database.
off-the-shelf	A type of software which is readily available and should be suitable for a large amount of people.

Glossary

primary key	The unique identifying value for records in a database.
programming language	A language used by a programmer to write a piece of software.
query language	A type of programming language used to work with databases. A typical example is SQL.
record	All of the data relating to one entity in a database.
smartphone	A mobile phone with a powerful processor that is capable of running applications and accessing the internet.
social network	Sites that allow people to communicate with others and share images, videos and other content.
software	The programs, applications and data in a computer system. Any parts of a computer system that aren't physical.
software model	Software that works with data to represent real or imaginary systems.
spreadsheet	A piece of software used to manipulate data, often used in modelling.