Database Design & Development - Summary

Analysis

Identify the end-user and functional requirements of a database problem that relates to the implementation at this level.

- **end-user** – what does the end user want to able to do with the database e.g. search for student data
- **functional requirements** – the tasks the database should perform/ data it should hold e.g. first name

Design

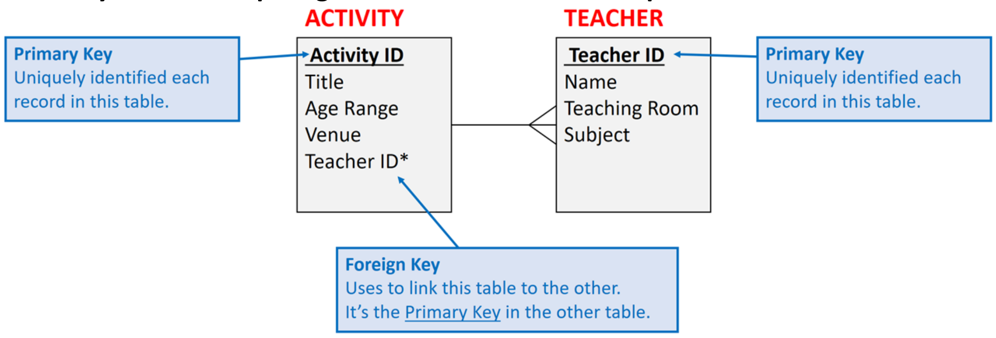
Describe and identify the implications for individuals and businesses of the Data Protection Act 1998.
The Data Protection Act (1998) is designed to ensure that personal data is held securely and is not shared with unauthorised third parties. Businesses storing personal data must ensure

- they have prior consent of data subject (the person the data is about) to store personal information
- the data is accurate and up-to-day
- the data used for limited, specifically stated purposes
- the data kept safe and secure

Describe and exemplify entity-relationship diagrams with two entities.

- **Entity** – a place where information about a person, thing or concept is collected – a table.
- **Attribute** – describe the facts, details or characteristic of an entity – a field.

An Entity-Relationship diagram shows the relationship that exists between two entities.


ACTIVITY
TEACHER
Primary Key
Uniquely identified each record in this table.
Activity ID
Title
Age Range
Venue
Teacher ID*
Teacher ID
Name
Teaching Room
Subject
Primary Key
Uniquely identified each record in this table.
Foreign Key
Uses to link this table to the other.
It's the Primary Key in the other table.

(remember that fields must be included in an ERD as well as indicating the PK and FK)
Describe and exemplify a data dictionary:
A data dictionary includes the name and description of the attributes in each entity within a database.

Table: Customers

Field	Key	Attribute Type	Required	Validation
Customer_ID	Primary Key	Number	Yes	Length = 3
Title		Text	Yes	Restricted Choice = Mr, Mrs, Miss, Ms,
First Name		Text	Yes	
Surname		Text	Yes	
Flight Date		Date	Yes	
First Class		Boolean	No	
Flight Number	Foreign Key	Text	Yes	Restricted Choice

Attribute Types:

- **Text** – a combination of text and number e.g. name, address, telephone number.
- **Number** – whole numbers and decimal numbers.
- **Date** – usually in the format DD/MM/YYYY
- **Time –** usually in the format 00:00 but can be 00:00:00
- **Boolean** – true or false

Validation – ensure data is allowable and sensible, makes it more difficult to make mistakes.

- **Range** – ensures data is between a lower and upper limit
- **Restricted Choice** – limits the user to a list of options.
- **Length Check** – restricts the number of characters that can be entered.
- **Presence Check** – ensures that the field is not left blank – it is required

Implementation

Implement relational databases with two linked tables, to match the design with referential integrity.
Referential integrity ensures that the foreign key value has a matching value in the corresponding primary key.

Describe, exemplify and implement SQL operations for pre-populated relational databases.
SQL standard for Structured Query Language, it is used to interrogate the data held in a database.

SELECT … FROM … – is used to select data from the database – `SELECT name, gender FROM student;`

… WHERE – where is used to ensure that only records that meet a specific condition are returned.

```
SELECT name, gender FROM student WHERE regClass = "1S1";
```

... ORDER BY – used to order the results in either ascending or descending order. `ORDER BY age ASC`

```
SELECT firstname, surname, gender
FROM student
WHERE regClass = "1S1"
ORDER BY surname ASC;
```

INSERT INTO – is used to add a record to a table.

```
INSERT INTO Pupil
VALUES (Ben, Simon, Male, 10/10/05, 1S1);
```

UPDATE – is used to update a current record held in the database. The WHERE identifies specific records.

```
UPDATE pupil
SET Reg Teacher = "Dr. G. Wilson"
WHERE Reg Teacher = "Mr E. Neil"
```

DELETE – delete a record from the database. No WHERE will mean all content is deleted from the table.

```
DELETE FROM pupil
WHERE firstName = "Eva" AND surname = "Forrest";
```

EQUI-JOIN – used to display result from two different tables. Requires Primary Key & Foreign Key to be linked

```
SELECT firstname, surname, instrument
FROM student, musicTution
WHERE student.studentCode = musicTution.studentCode AND year < 4:
```

List the name and instrument played by all pupils in S1-3

Operators

- **AND** – both conditions must be true - `WHERE regClass = "1S1" AND gender= "female";`
- **OR** – either condition must be true - `WHERE year = "1" OR year= "2";`
- **<** - less than - `WHERE age < 18;`
- **>** - greater than - `WHERE year > 3;`

Read and explain code that makes use of the above SQL.

Testing

Describe and exemplify testing SQL operations work correctly at this level

- When testing queries, it is good practice to document expected results and actual results.
  A comparison can then be made, with testing passed if the expected result and actual result is the same.

Evaluation

Evaluate solution in terms of fitness for purpose & accuracy of output

- A database is deemed fit for purpose if it meets the requirements determined at the analysis stage. If the database is not fit for purpose, it will be necessary to revisit previous phases of the development process.

SQL Quick Guide

SQL dates are stored in the format yyyy/mm/dd

SELECT Statements

SELECT statements are used to query a database and return some values.

- Specify the fields you want to display: **SELECT fieldname1, fieldname2**
- Name the tables that the fields are in: **FROM tablename**
- The criteria for your search: **WHERE fieldname1 = 'value'**
- Any sort order that you want to apply: **ORDER BY fieldname1 ASC/DESC**

```
SELECT name, age
FROM pupil
WHERE age <18
ORDER BY name ;
```

When searching for text values you must use 'speech marks'
You can use logical conditions such as AND, OR and NOT to create more complex queries.

DELETE Statements

DELETE statements delete records from a table

- Name the table where the records are to be deleted: **DELETE fieldname**
- Which tables that the fields are in: **FROM tablename**
- The criteria for your search: **WHERE fieldname1 = "value"**

```
DELETE FROM Course
WHERE courseID='BMX05';
```

If you don't specify criteria then every record in the table will be deleted!

INSERT Statements – Entering values for every field

INSERT statements can be used to add new records into the database.

- Specify the tablename to insert into: **INSERT INTO Instructor**
- Specify the values to be inserted: **(5, 'D Thomas', '1985/11/30', 5)**

```
INSERT INTO Instructor
VALUES  (5, 'D Thomas', '1985/11/30', 5);
```

INSERT Statements – Only entering values for some fields

Sometimes you may not be adding a value for every single field.

- Specify the tablename to insert into: **INSERT INTO Instructor**
- Specify the fields you are entering data for: **(InstructorID, Name)**
- List the values to be inserted: **(5, 'D Thomas')**

```
INSERT INTO Instructor (InstructorID, Name)
VALUES (5, 'D Thomas');
```

UPDATE Statements

UPDATE statements will edit existing data in the table.

- Name the table where the records are: **UPDATE tablename**
- Set the new values: **SET fieldname = value1, fieldname = value2**
- Criteria if there are any: **WHERE fieldname = fieldvalue**

```
UPDATE Course
SET Date = '2017/12/11'
```

EQUI-JOINS

An equi-join is when two matching columns from related tables are selected (joined) using the primary key and its matching foreign key from the tables.

```
SELECT Instructor.Name, Course.Title, Course.Date
FROM Instructor, Course
WHERE Instructor.InstructorID = Course.InstructorID AND Instructor.Name = "John Smith" ;
```