FACTFILE:
GCE SOFTWARE SYSTEMS DEVELOPMENT

(v) DATABASE CONCEPTS

Rewarding Learning

Database Concepts

Learning outcomes

Students should be able to:
- examine the concept of a database and associated terminology;
- explain the nature of databases:
  - Relational model;
  - Hierarchical models;
  - Network;

Content

Files/Databases
Files previously considered in unit AS-1, 'Introduction to Object Oriented Development', introduced storage concepts suitable for very basic applications. In the real world, using files for data storage causes many problems. Some of these problems are listed below:
- Different applications within an organisation may have duplicated data in files often resulting in inconsistency impacting on data integrity;[1]
- Sharing data between applications or across organisations may necessitate the creation of programs to merge files which store data in different file formats;
- Changing the file format (e.g. adding another item of data to be stored) will require changes to the programs that read from and write to the files;
- Reporting requires programming expertise and ad hoc requests may be difficult to satisfy.

A Database:
- has a single repository of data which minimises data duplication/redundancy[2]. Hence storage space is minimised and the risk of data inconsistency is reduced. It enables easier data sharing allowing more users to derive more meaningful information as they can access more data;
- is self-describing in that it stores not only data but also information about the structure of that data (e.g. names of data items, data types and the range of values possible), also referred to as meta-data. This allows a certain degree of data independence as minor modifications can be made to the database structure without requiring changes to connecting applications. Maintenance is easier;
- is designed to meet the needs of the organisation. The database schema is seen as a data model of that part of the real world in which the organisation is interested.

[1]Data Integrity refers to the validity and consistency of the data. Rules or constraints can be established about the data that is allowed to be stored in the database. If data is duplicated then updates to one piece of data (e.g. a customer's address) if not performed on the duplicate will lead to inconsistent data (e.g. two different addresses are stored in two separate places for one customer and one of them is updated; which one is valid?).
[2]Duplicated data may be redundant: Redundant data may be deduced from other data in the database; it may be deleted without information being lost. Sometimes we need duplicate data to form relationships; this is not redundant data as it provides information about the relationship that should not be deleted.

A Data Base Management System (DBMS) contains a collection of programs to:

- define the Database using a Data Definition Language (DDL), for example the CREATE TABLE command in SQL (Structured Query Language). The definition of the database is stored separately from the actual data;

- Insert, Update, Delete and Retrieve Data using a Data Manipulation Language (DML), for example the SELECT command in SQL. The DBMS facilitates easier access to the database. This improves the productivity of both users and programmers by eliminating the need to write complex code. Graphical tools may also be provided;

- manage and control data access using tools for:
  - **Security:** Integrating data and allowing shared access can make data more vulnerable to access by unauthorised users. Groups, user names and passwords can be created to restrict access;
  - **Data Integrity:** Erroneous data, inserted by one user will affect other users that share the data. Constraints (rules that the database is not permitted to violate) can be added to maintain the validity and consistency of data;
  - **Concurrency:** When two people access the same data at the same time, called 'concurrent access', problems can arise with updating;
  - **Backup and Recovery:** Database failure can have a serious impact on an organisation. The provision of regular back-ups and a recovery procedure is imperative.

</> Data Models

There are three main data models[3] based on groups of records with a fixed format and a fixed number of fields. The **Hierarchical** model (e.g. IBM's Information Management System, IMS) and the **Network** model (e.g. Computer Associates' IDMS/R - Integrated Database Management System -; CODASYL - Conference/Committee on Data Systems Languages - approach) were developed in the 1960s and 1970s. They are considered legacy systems and have many links to traditional file processing. The **Relational** model was proposed in 1970 by the mathematician Codd. Most commercial databases in use today are relational (e.g. SQL and MySQL).

Relational Model:

This is based on the simplest of structures, the **table/entity** (formerly referred to as a **relation**), composed of rows and columns. A table is essentially a collection of **records** that correspond to the **rows** of the table. Each record is distinct. Each column of the table corresponds to a field/attribute. Each **column** has a unique name and **must** hold the same data type. Each cell of the table contains exactly one single value. This model is shown in Table 5.1 below.

Table: Composed of rows and columns, has unique name and information about objects to be represented.

Primary Key: Set of fields that uniquely identifies each record.

Field: Column of table with unique name. Values have the same data type. Order is irrelevant.

Record: Row of the table Each row is distinct. No duplicates are allowed, Order is irrelevant

Cell: Contains one single value

Student ID	Firstname	Surname
S10	Fiona	Whyte
S2	John	Black
S11	David	Brown
S23	Mary	Black
S1	Fiona	O'Hara

Table 5.1 – Relational Database Model

[3]Readers may also be aware of other data models: Object and Object–Relational. DBMS products based on these models more easily support the persistent storage of objects created in object oriented programs using Java or C#. 'Big Data' used in Data Analytics is another example.

To uniquely identify each record, a field or set of fields is used as a primary key. A primary key has no identical values in any two records of the table. In the example above the primary key is StudentID. No field of a primary key can be null (unknown/ absent value).

If more than one field is used as the primary key it is said to be a composite primary key. In the MARK table shown opposite, the primary key is composite consisting of both ExamCode and StudentID. Both of these fields are required to identify a student's mark.

Tables are linked using one-to-many relationships. The primary key of the parent table is included as a foreign key in the child table. This forms the one-to-many relationship. The first table holds information about the student where the StudentID is the primary key and uniquely identifies each student. The second table holds the marks attained over a range of exams, for each student. The composite key of the StudentID and the ExamCode is necessary to uniquely identify a mark.

The marks for the student with an ID of S10 (Fiona Whyte) can be determined from this primary-foreign key link (Ex56:66; Ex34: 30).

Referential Integrity must be established between tables. This will ensure that data in the child table is consistent with data in the parent table. In order for a mark to be entered for a particular student in the MARK table, the corresponding student record must already exist in the STUDENT table.

ExamCode	StudentID	Mark
Ex10	S2	50
Ex34	S10	30
Ex34	S2	89
Ex56	S10	66
Ex56	S2	40

Table 5.2 – Mark Table

Tables are often represented by their table name and a list of fields enclosed in brackets, with the primary key underlined. Foreign keys can be represented with an '*'. For example:
STUDENT (StudentID, FirstName, Surname)
MARK (ExamCode, StudentID*, Mark)

Data is retrieved from sets of tables by the use of relational algebra. Specific columns and/or rows can be selected and tables joined together. Only one-to-many and one-to-one relationships can be formed using this primary/foreign key mechanism. Many-to-many relationships can only be represented in the relational model by decomposition into two one-to-many relationships and the insertion of a new table in the middle. The new table is referred to as a link table.

Users do not need to know how the data is physically stored in this model. Users perceive the database as a set of interconnected tables. In reality the storage may be implemented using a variety of methods. The DBMS manages the physical storage for the users.

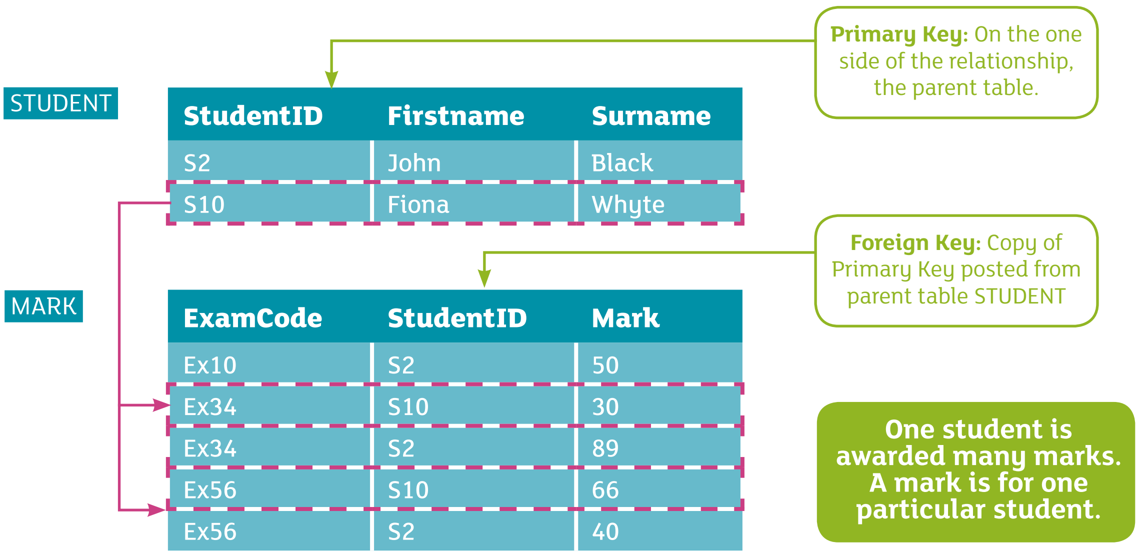

Figure 5.1 – This shows how the STUDENT table is related to the MARK Table

Hierarchical Model

The data in this model is organised in a tree (actually an upside down tree) structure. At the top/root of the inverted tree there is only one record type. Each record type can be the parent of many different types of child record. Every child must have a parent, but only one parent is ever allowed. This is obviously very restrictive and really only suits data that can be modelled in a hierarchical fashion (from one-to-many). Many-to-many relationships cannot be implemented without major difficultly and compromising the model.

Network Model

In a network model a record can have many owners of a different record type. This allows the network model to be much more flexible (allowing many more connections between records) in representing data structures. It is easier to support many-to-many relationships such as that between Course and Student (after decomposition, as in the relational model).

The Hierarchical model is really like a restricted network model. In the hierarchical model each record can have only one parent.

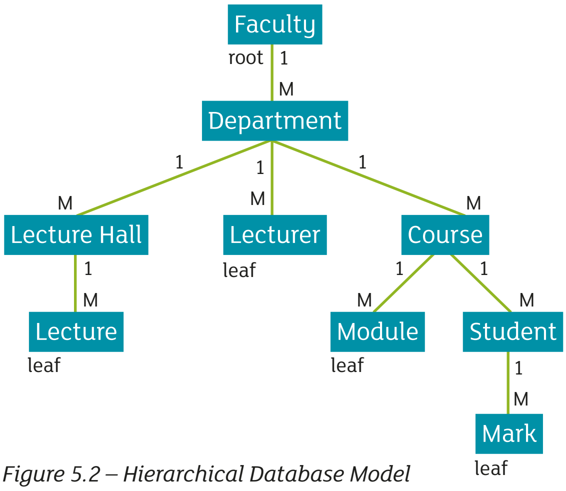

Figure 5.2 – Hierarchical Database Model

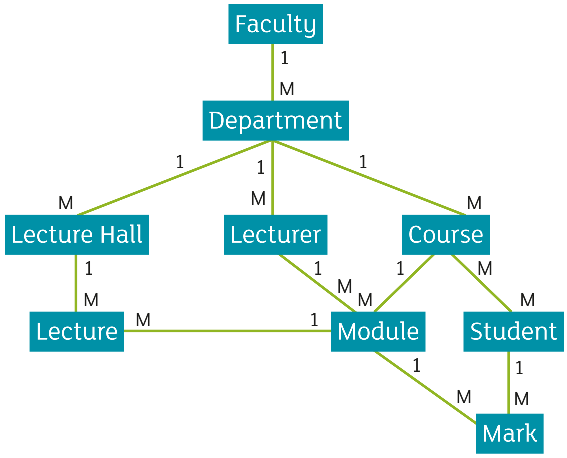

Figure 5.3 – Network Database Model

Comparison between the different database models

- All of the models may refer to data being stored in files, records and fields (although the relational model tends to use the terminology of tables, records and fields).
- Network and hierarchical models often use pointers to implement relationships; the relational model implements relationships using a primary-foreign key mechanism.
- Both the hierarchical and the network model require knowledge of the physical storage of the records in a file or memory. Accessing records requires navigation of this physical structure from one record to another. The physical storage of data is of no relevance in the relational model (which is perceived to be tables of data). The DBMS handles all of this for the user and no complex code is needed.
- The relational model enables users to specify what is required using queries. The relational model is easier for users to adapt their business requirements.
- *Many-to-many* relationships can be supported by simple decomposition (into *one-to-many* relationships) in both the relational and network models but not in the hierarchical model.

Questions

1 Give two advantages of a database approach compared to traditional file processing.

Advantage 1

....

....

Advantage 2

....

....

2 Describe **two** important features of a database.

Feature 1

....

....

Feature 2

....

....

Questions

3 Describe two advantages a DBMS may offer and why they are important.

Advantage 1

....

Importance

....

....

....

Advantage 2

....

Importance

....

....

....

Questions

4 Explain the following terms:

Term	Explanation
Database	
DBMS	
Data independence	
Data inconsistency	
Concurrency	
Data integrity	
Data duplication	
Data redundancy	

Table 5.3

Questions

5 A relational database includes the tables BATCH, INGREDIENT and MATERIAL, shown below.

BATCH

BatchNo	DateofManufacture	NoProduced	PassQualityCheck	ProductName
1	2014-12-02	25	Y	Night Night
2	2014-01-08	25	Y	Skin Cleanser
3	2014-02-13	20	Y	Skin Soother

INGREDIENT

BatchID	MaterialID	Weight
1	1	1801.04
1	2	3610.10
1	3	310.10
1	4	300.02
1	5	300.06
2	1	1800.00
2	2	3601.00
2	3	300.56
2	8	310.06
2	13	300.06
3	1	1800.01
3	2	3600.11
3	3	300.05
3	4	300.10
3	6	300.07

Questions

5 MATERIAL

MaterialID	CommonName	DateReceived	QuantityDelivered	CostPerUnit
1	Citric Acid	2014-01-01	3000.00	0.15
2	Baking Soda	2014-01-01	9500.00	0.23
3	Almond Oil	2014-01-01	500.00	0.34
4	Chamomile Essential Oil	2014-01-01	50.00	0.45
5	Lavender Essential Oil	2014-01-01	50.00	0.35
6	Goat's Milk Powder	2014-01-01	100.00	0.20
7	Shea Butter	2014-01-01	500.00	0.15
8	Epsom Salts	2014-01-01	450.00	0.28
9	Seaweed Extract	2014-01-04	450.00	0.28
10	Green Clay	2014-01-04	450.00	0.28
11	Citric Acid	2012-04-20	3000.00	0.15
12	Baking Soda	2012-04-23	9500.00	0.23
13	Tea Tree Essential Oil	2014-08-30	75.00	0.38

With reference to the tables, give examples of each of the following terms:

Table **Record** **Field** **Primary Key**
Composite Primary Key **Foreign Key** **Referential Integrity**

Explain what would happen when the following records are added to the INGREDIENT table.

Record 1
BatchNo=2
MaterialID=3
Weight =45.2

Explanation of result:

....

....

....

Questions

5

Record 2
BatchNo=4
MaterialID=13
Weight =110.5

Explanation of result:

....

....

....

Record 3
BatchNo=3
MaterialID=15
Weight =25.2

Explanation of result:

....

....

....

Record 4
BatchNo=3
MaterialID=9
Weight =200.5

Explanation of result:

....

....

....

Questions

6 Complete the table below comparing the hierarchical, network and relational data models.

	Hierarchical	Network	Relational
Example of commercial database			
How data and relationships are represented		Network. A record may have many owners of a different type.	
Implementation of *one-to-many* relationships	Pointers often used.	Pointers often used.	
Relationships supported	*One-to-many*	*One-to-many* and *many-to-many* only after decomposition.	
Accessing Data	Physically traverse the records often using pointers to location in file or memory.		
Expertise required	Expert knowledge required because of need to use physical location of data for storage and queries.		

Table 5.3 – Comparison of Different Data Models