OCR Computer Science A Level

1.3.2 Databases

Intermediate Notes

Specification:

1.3.2 a)

- Relational Database
- Flat File
- Primary Keys, Foreign Keys, Secondary Keys
- Entity relationship modelling
- Normalisation
- Indexing

1.3.2 b)

- Methods of capturing, selecting, managing, and exchanging data

1.3.2 c)

- Normalisation

1.3.2 d)

- SQL

1.3.2 e)

- Referential Integrity

1.3.2 f)

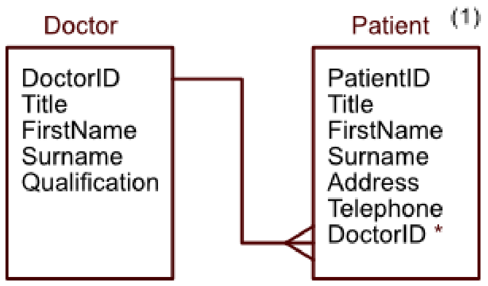
- Transaction processing
- ACID (Atomicity, Consistency, Isolation, Durability)
- Record locking
- Redundancy

Relational Database

Relational Databases

An entity is an item of interest about which information is stored. A relational database is a recognises the differences between entities by creating different tables for each entity.

The diagram on the right is an entity relationship model which shows two entities: Doctor and Patient. DoctorID is the attribute linking the two tables together. Attributes are characteristics of the entity; these are categories about which data is collected.



Flat File

A flat file is a database that consists of a single file.
The flat file will most likely be based around a single entity and its attributes. Flat files are typically written out in the following way:

Entity1(<u>Attribute1</u>, Attribute2, Attribute3 …)

The example in the table shows the entity Car. Age and Price are attributes of each car. For this example, the description would be laid out as:

Car(<u>CarID</u>, Age, Price)

Car		
CarID	**Age**	**Price**
Car1	5 years	£1,500
Car2	2 years	£2,400

(2)

Primary Key

A primary key is a unique identifier for each record in the table. In example (2), the unique identifier is the CarID as this is always different for each row in the table. The primary key is shown by underlining it.

Foreign Key

A foreign key is the attribute which links two tables together. The foreign key will exist in one table as the primary key and act as the foreign key in another. In example (1), DoctorID is the foreign key in the Patient table. The foreign key is shown using an asterisk.

<u>Secondary Key</u>

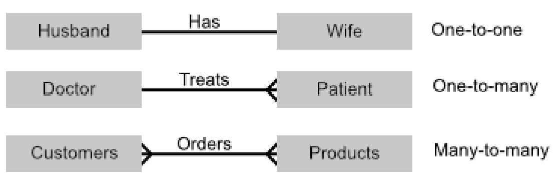
A secondary key allows a database to be searched quickly. Looking at example (1), the patient is unlikely to remember their patientID but will know their surname. Therefore, a secondary index (secondary key) is set up on the surname attribute. This makes it possible to order and search by surname which makes it easier to find specific patients.

<u>Entity Relationship Modelling</u>

Tables can have different kinds of relationships:

- One-to-one: Each entity can only be linked to one other entity, such as the relationship between a husband and wife. The Husband entity can only be associated with one Wife entity and vice versa.
- One-to-many: One table can be associated with many other tables, such as a mother having multiple children. Similarly, multiple child entities can be linked to the same mother entity.
- Many-to-many: One entity can be associated with many other entities and the same applies the other way round. An example is students and courses - each student can enrol in more than one course and each course can have more than one student.

One-to-one relationships are demonstrated using a single line used to connect two entities. A one-to-many relationship will have a branch on one side, while a many-to-many relationship has branches on both sides.



<u>Normalisation</u>

The process of coming up with the best possible layout for a relational database is called normalisation.

Normalisation tries to accomplish the following things:

- No redundancy (unnecessary duplicates).
- Consistent data throughout linked tables.
- Records can be added and removed without issues.
- Complex queries can be carried out.

There are three types of normalisation:

<u>First Normal Form</u>

There must be no attribute that contains more than a single value.

<u>Second Normal Form</u>

A database which doesn't have any partial dependencies and is in first normal form can be said to be in second normal form. This means that no attributes can depend on part of a composite key.

Third Normal Form

If the database is in second normal form and contains no non-key dependencies, it is in third normal form. A non-key dependency means the attribute only depends on the value of the primary key and nothing else.

Indexing

Indexing is a method used to store the position of each record ordered by a certain attribute. This is used to look up and access data quickly. The primary key is automatically indexed.

Handling Data

Capturing Data

Data needs to be input into the database and there are various ways of doing this. The chosen method is always dependent on the context. For example, if pedestrians are participating in a survey, their responses will need to be manually entered.

Data is also captured when people pay cheques. Banks scan cheques using Magnetic Ink Character Recognition (MICR). Optical Mark Recognition (OMR) is used for multiple choice questions on a test. Other forms use Optical Character Recognition (OCR).

Selecting and Managing Data

Selecting the correct data is an important part of data preprocessing. This could involve only selecting data that fits a certain criteria to reduce the volume of input. Collected data can be managed using SQL to sort, restructure and select certain sections.

Exchanging Data

Exchanging data is the process of transferring the data that has been collected. One common example of this is EDI (Electronic Data Interchange).

SQL

SQL stands for Structured Query Language and is a declarative language used to manipulate databases. SQL enables the creating, removing and updating of databases.

Synoptic Link

Declarative programming language is a programming paradigm

Programming paradigms are covered in **1.2.4 Types of Programming Language**

SELECT, FROM, WHERE

The SELECT statement is used to collect fields from a given table and can be paired with the FROM statement to specify which table(s) the information will come from. The WHERE statement can be used in conjunction to specify the search criteria.

Movie				
MovieID	MovieTitle	MovieCompany	DatePublished	DirectorName
M0001	Howdy Partner!	Cowboys Inc	04/21/2001	James
M0002	Okay Samantha, just leave.	Sadboys Inc	04/21/2001	Joseph
M0003	Bye Bye Bucky.	Cowboys Inc	05/12/2004	Jeremy
M0004	My wife left me for my dog...	Sadboys Inc	05/14/2004	James
M0005	Cars, Girls, and Money.	Rappers ltd	06/21/2012	James
M0006	Water Bottle Sadness	Sadboys Inc	08/12/2015	Jeremy

For example:

```
SELECT  MovieTitle, DatePublished
FROM Movie
WHERE DatePublished BETWEEN #01/01/2000# AND #31/12/2005#
ORDER BY DatePublished
```

This will produce the following result:

MovieTitle	DatePublished
Howdy Partner!	04/21/2001
Okay Samantha, just leave.	04/21/2001
Bye Bye Bucky.	05/12/2004
My wife left me for my dog...	05/14/2004
Cars, Girls, and Money.	06/21/2012
Water Bottle Sadness	08/12/2015

ORDER BY

The ORDER BY part of the code specifies whether you want it in ascending or descending order. The example below orders selected data in descending order:

```
ORDER BY DatePublished Desc
```

JOIN

JOIN provides a method of combining rows from multiple tables based on a common field between them. The example below shows the joining of two tables, Movies and Directors.

```
SELECT Movie.MovieTitle, Director.DirectorName, Movie.MovieCompany
FROM Movie
JOIN Director
ON Movie.DirectorName = Director.DirectorName
```

CREATE

The CREATE function allows you to make new databases, as shown below:

```
CREATE TABLE TableName
(
Attribute1          INTEGER NOT NULL, PRIMARY KEY,
Attribute2          VARCHAR(20) NOT NULL,
…
)
```

The following details about each attribute must be specified:

- Whether it is the primary key
- Its data type
- Whether it must be filled in (‘Not Null’)

Synoptic Link

Data Types are names given to different formats of data.

Data types are covered in **1.2.1 Data Types**.

Data Types:

- CHAR(n): this is a string of fixed length n
- VARCHAR(n): this is a string of variable length with upper limit n
- BOOLEAN: TRUE or FALSE values
- INTEGER/INT: integer
- FLOAT: number with a floating decimal point
- DATE: the date in the format Day/Month/Year
- TIME: the time in the format Hour/Minute/Second
- CURRENCY: sets the number as a monetary amount

ALTER

ALTER is used to add, delete or modify the columns in a table

Adding a column:

```
ALTER TABLE TableName
ADD AttributeX and their dataTypes
```

Deleting a column:
```
ALTER TABLE TableName
DROP COLUMN AttributeX
```

Modifying the data type of a column:
```
ALTER TABLE TableName
MODIFY COLUMN AttributeX  NewDataType
```

INSERT INTO

This is used to insert a new record into a database table.

For example:
```
INSERT INTO (column1, column2, …)
VALUES (value1, value2, …)
```

UPDATE

This is used to update a record in a database table.

For example:
```
UPDATE TableName
SET column1 = value1, column2 = value2 …
Where columnX = value
```

DELETE

This is used to delete a record from a database table.

For example:
```
DELETE FROM TableName
WHERE columnX = value
```

Referential Integrity

Referential integrity is the process of ensuring consistency. This ensures that information is not removed if it is required elsewhere in a linked database.

Transaction Processing

A transaction is defined as a single operation executed on data.

ACID (Atomicity, Consistency, Isolation, Durability)

Atomicity:

- A transaction must be processed in its entirety or not at all,

Consistency:

- A transaction must maintain the referential integrity rules between linked tables,

Isolation:

- Simultaneous executions of transactions should lead to the same result as if they were executed one after the other,

Durability

- Once a transaction has been executed it will remain so regardless of the circumstances surrounding it, such as in the event of a power cut.

Record Locking

The process of preventing simultaneous access to records in a database is called record locking and it is used in order to prevent inconsistencies or a loss of updates. While one person is editing a record, this 'locks' the record so others cannot access the same record. The biggest problem with this is deadlock, described below.

- *User 1 accesses Customer 1's record and as a result locks Customer 1's record. Simultaneously User 2 accesses Customer 2's record and as a result locks Customer 2's record.*
- *Now User 1 tries to access Customer 2's record, and User 2 tries to access Customer 1's record.*
- *User 1 waits for Customer 2's record to be free and User 2 waits for Customer 1's record to be free and as they are both waiting, there is no progress causing a deadlock.*

Redundancy

There is some information that people and companies cannot afford to lose. Redundancy is the process of having one or more copies of the data in physically different locations. This means that if there is any damage to one copy the others can be recovered.