AQA Computer Science A-Level

4.10 Fundamentals of databases

Concise Notes

Specification:

4.10.1 Conceptual data models and entity relationship modelling:

Produce a data model from given data requirements for a simple scenario involving multiple entities

Produce entity relationship diagrams representing a data model and entity descriptions in the form: Entity1 (Attribute1, Attribute2, ...)

4.10.2 Relational databases:

Explain the concept of a relational database

Be able to define the terms:

- attribute
- primary key
- composite primary key
- foreign key

4.10.3 Database design and normalisation techniques:

Normalise relations to third normal form

Understand why databases are normalised

4.10.4 Structured Query Language (SQL):

Be able to use SQL to retrieve, update, insert and delete data from multiple tables of a relational database

Be able to use SQL to define a database table

4.10.5 Client server databases:

Know that a client server database system provides simultaneous access to the database for multiple clients

Know how concurrent access can be controlled to preserve the integrity of the database

Data models

- When creating a database, you might be given data requirements from which you need to produce a data model
- A data model is an abstract model of which things to store and what information about them should be recorded

Entities and attributes

- An entity is a thing about which data is to be stored
- Attributes are characteristics or other information about entities
- Databases are formed of tables which are used to store multiple entities
- Each entity usually has its own row in a table and fields of that row hold the entity's attributes

Entity identifiers

- When creating a database, it's important to ensure that each entity has an unique identifier
- An entity identifier is an attribute given to each entity which is unique within that table
- Some database tables have multiple attributes which are combined to form the table's entity identifier

Entity description

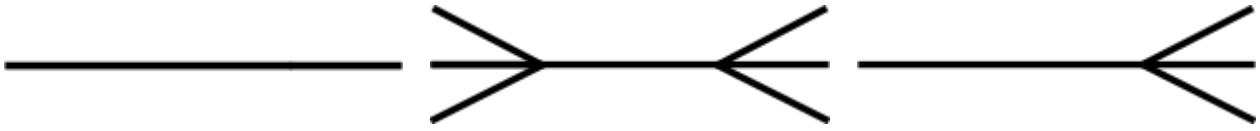
- An entity description can be used to describe how information is to be stored about entities
- The name of the table is shown outside of brackets which contain each of the entity's attributes separated by commas
- Underlining can be used to identify the attribute or attributes which form the table's entity identifier

Relational Databases

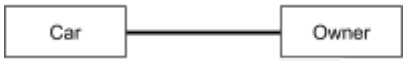
- The tables in a database can be related to each other, linked by common attributes
- There are three possible degrees of relationship between tables in a database:
    - one-to-one
    - many-to-many
    - one-to-many

Entity relationship diagrams

- Are used to graphically represent the relationships between tables in a database
- Tables are shown as rectangles and are joined by lines which can represent different types of relationship



One-to-one Many-to-many one-to-many



Each car has one owner, and each owner has one car (this example assumes that nobody owns multiple cars).

Each car has many passengers. Each passenger sits in one car.

Each driver can drive many different cars. Each car is driven by many different drivers.

Primary and foreign keys

- A primary key is an attribute that provides a unique identifier for every entity in a database table
- When tables are linked by a shared attribute, the attribute must be a primary key in one table and is called a foreign key in the other
- A foreign key is an attribute in a table which is the primary key in another, related, table
- It is possible to combine attributes to form what is called a composite primary key

Many-to-many relationships

- When linking many-to-many relationships, a new table has to be created
- This new table is called a link table

Database Normalisation

- Databases are normalised so that they can be efficient without any compromise to the integrity of their data
- Normalising databases involves ensuring that entities contain no redundant or repeated data
- A database that has been normalised allows for faster searching and sorting that an unnormalised database
- Normalised databases are easier to maintain than unnormalised databases
- Duplication of data is minimised and data consistency is improved
- Normalisation helps to reduce the number of update, insertion and deletion anomalies

First normal form

- A database in first normal form:
  - must not contain any repeating attributes
  - can be referred to as having atomic data - no single column contains more than one value

Second normal form

- A database in second normal form must:
  - satisfy first normal form
  - have any partial key dependencies removed
- A partial key dependency occurs in databases with composite primary keys when a non-key attribute doesn't depend on the whole of the composite key

Third normal form

- A database in third-normal form must:
  - conform to second normal form
  - have no non-key dependencies
- A database that meets third normal form can be described as follows:

All non-key attributes depend on the key, the whole key and nothing but the key

Structured Query Language (SQL)

- Is a language used with databases
- Is easy to learn and use
- Is a declarative language, meaning that the programmer describes the result that's required rather than describing the process which should be followed
- Has four main commands: SELECT, UPDATE, INSERT and DELETE

The SELECT command

- Used for retrieving data from a database table
- Commands take the following form:

```
SELECT <attribute> FROM <table> WHERE <condition> ORDER BY <ASC/DESC>
```

- The ORDER BY clause is optional

The UPDATE command

- Used in databases for modifying the attributes of an existing entity
- Commands take the form:

```
UPDATE <table> SET <attribute> = <value> WHERE <attribute> = <value>
```

The DELETE command

- Used for removing entities from a database
- The commands take the following form:

```
DELETE FROM <table> WHERE <condition>
```

The INSERT command

- Used to add new records to an existing table
- The command usually takes the form

```
INSERT INTO <table> (<column1>, <column2>, …) VALUES (<value1>, <value2>, …)
```

but can be simplified to

```
INSERT INTO <table> VALUES (<value1>, <value2>, …)
```

when all of the columns in the table are being used in the correct order.

Wildcards

- Can be used in SQL commands to specify any possible value
- Wildcards are usually notated with an asterix

Defining a table with SQL

- SQL can be used to make new database tables with the CREATE command
- This command specifies the name of the new table, its attributes and their data types
- Also specified are entity identifiers like primary and secondary keys
- The CREATE command creates an empty table
- New records can be added using the INSERT command

SQL Data Types

Data type	SQL	Description
Fixed length string	`CHAR(size)`	A string with the number of characters specified by size
Variable length string	`VARCHAR(size)`	A string with any number of characters up to the number specified by size
Integer	`INT(size)`	A whole number stored using the number of bits specified by size
Number with fractional part	`FLOAT(size, precision)`	A number stored using the number of bits specified by size with digits after the decimal point up to the number specified by precision
Date	`DATE`	A date in the format YYYY-MM-DD
Date and time	`DATETIME`	A date and time combined in the format YYYY-MM-DD HH:MM:SS
Time	`TIME`	A time in the format HH:MM:SS
Year	`YEAR`	A year in one of the two formats YY or YYYY

Client server databases

- Provide simultaneous access to a database for multiple clients
- Issues rarely arise when two users are requesting access to different, unrelated fields in a database
- When different users attempt to access the same field at the same time, a problem known as concurrent access occurs
- Concurrent access can result in database updates being lost if two users edit a record at the same time
- Concurrent access can be managed with the use of record locks, serialisation, timestamp ordering and commitment ordering

Record locks

- When a record is accessed by one user, it is immediately locked to other users until the first user has finished using it
- Other users are blocked from accessing or modifying the content of a field until it has been unlocked

Serialisation

- Rather than locking a field, requests from other users are placed in a queue
- Once the first user has finished using the field, the next command in the queue is executed and so on

Timestamp ordering

- When multiple commands are sent to the same field in a database, each is assigned a timestamp which marks the point in time at which the command was initiated
- Commands are carried out on the field in the order of their timestamps

Commitment ordering

- When a database uses commitment ordering, an algorithm is used to work out an optimum order in which to execute commands for the same field
- This algorithm will take into account the impact of commands on other parts of the database
- This helps to prevent issues from occurring with the database