Indexing

As motioned when talking about the Primary Key, these are automatically indexed, but should we have another column we wish to index specifically we can do this in two ways

First way is to index your table at the time of creation. This is accomplished by including the Keyword INDEX or UNIQUE in the say way we defined a Primary Key. Here is an example

```
CREATE TABLE invoice {
    invoice_id   INT          NOT NULL AUTO_INCREMENT,
    purchase_id  INT          NOT NULL,
    item_name    VARCHAR(20),
    item_desc    VARCHAR(50),
    value        DECIMAL(7,2) NOT NULL,
    PRIMARY KEY (invoice_id),
    INDEX purchase_id (purchase_id),
};
```

If we have already created the table there is no problem adding the index at a later date. We do this by using the CREATE keyword but instead of creating a table we create an index on a table. Here the syntax is

```
CREATE [UNIQUE] INDEX index_name ON tbl_name (col_name, ...);
```

Using the example above as if we hadn't created the index we would write

```
CREATE INDEX purchase_id
ON invoice (purchase_id);
```

Just like with a table we can also remove the index with the DROP keyword

```
DROP INDEX index_name ON table_name;
```

The options keyword UNIQUE allows us to restrict the values stored in the index to a unique number and as such also restricts the entry into the column or columns defined by the index to also be unique.

The use of indexes within tables should be carefully considered as they can affect the performance of your database. To some extent this is removed from your hands by the RDBMS optimiser as though you may define indexes, it is the decision of the optimiser if to implement them or not. This said, well thought out indexing in designing your database can be very beneficial in the performance of your database.