# Task 2 Critical Discussion

## Object-Relational

The object-relational model was chosen as it has a lot more features compared to a relational model. For example, use of types and inheritance. Creating objects is useful, as it allows these features to be used, and can help prevent copying code. More code reuse is good, and can reduce processing times. Objects are also easier to maintain than tables, which is ideal for the future after the database is initially created.

A relational model has several limitations which can be solved using an object-relational model, such as no inheritance or types.

## Types

Using object types allows data to be condensed into one type. For example,

Rather than have ‘street’,’city’,’postcode’

It is possible to have oAddress(‘<street>’,’<city>’,’<postcode>’)

Eg. oAddress(‘Colinton’,’Edinburgh’,’EH54 3NS’)

This makes the code much more readable and condensed.

## Inheritance

Inheritance is very useful in object-relational databases. It allows code re-usage and makes it unnecessary to retype fields that have been used elsewhere that share a common field.

For example,

Both an employee and a customer have first names, surnames, addresses etc.

Without inheritance, each table would have to both have these fields

But with inheritance, it is possible to have an abstract ‘personType’ object, which has these fields, which the customer and employee can use

This is also very useful for futureproofing – if a new type of person is introduced, it is a simple matter of inheriting these fields

## Encapsulation

Using an object-relational database uses encapsulation. This means that fields that are used by multiple types do not interfere with each other.

For example,

If a Savings Account has a certain value for ‘salary’ and a Current Account also has a certain value for ‘salary’ they will not interfere with each other, even though they are both inheriting the field from accType