يتم التعامل مع الوراثة بواحد من الثلاث حالات:

* **في حالة ال table per hierarchy(tbh) pattern** يتم التعامل مع ال derived classes التي ترث من base class من خلال التعريف الصريح له من خلال DbSet<className>

وبالتالي EF Core هيعمل جدول واحد بيمثل كل الوراثة، وفيه أعمدة تجمع بين الكلاس الاب و الأبناء مع إضافة عمود (descriminator) لتمييز مصدر الخاصية و جعل خصائص الأبناء nullable. لأنه لو جعلهnot nullable كل صف في كلاس الاب هيحتاج قيمة لخصائص الابناء رغم إنه ملوش أي خاصية منهم أصلا وده مستحيل.

* **في حالة table per type mapping pattern** كل Class في الـ inheritance hierarchy بياخد جدول منفصل في قاعدة البيانات مع ربط جداول ال derived classes ب foreign key لجدول ال base class. و ده باستخدام modelBuilder.Entity<className>().ToTable()

او يتم استخدام
modelBuilder.Entity<className>().UseTptMappingStrategy()
لكل base class و سيتم معرفة ال derived classes تلقائيا

* **في حالة table per mapping hierarchy(TPC) mapping pattern**

يعني كل كلاس base أو derived بيبقى ليه جدول مستقل في قاعدة البيانات وكل جدول يحتوي جميع خصائص الكلاس نفسه بالإضافة الى خصائص الكلاس الاب وبالتالي مش هيكون في foreign keys او joins و بالتالي هو بيحل مشكلة الأداء في طريقة tpt ولكن في مقابلها تظهر مشكلة تكرار الخصائص في الجداول. و لاستخدام هذه الطريقة :

modelBuilder.Entity<className>().UseTpcMappingStrategy()