1️⃣ Clustered Index: Composite PK vs Column PK

- **Clustered Index** = طريقة ترتيب البيانات في الجدول نفسه.
- **PK عادي (single column)** → زي EmployeeId (auto increment).
  - أسرع في البحث المباشر.
  - أبسط في الصيانة.
- **Composite PK** زي جدول → (أكتر من عمود) OrderDetails (OrderId, ProductId).
  - يمنع التكرار بين العمودين.
  - بيبطّأ عمليات البحث لو queries معقدة.
- **EF Core**:
  - Single PK → builder.HasKey(e => e.Id);
  - Composite PK → builder.HasKey(e => new { e.OrderId, e.ProductId });

📌 نصيحة: استخدم Composite PK بس في جداول الربط.

2️⃣ Lazy vs Eager Loading

- **Lazy**:
  - لما تجيب Employee مش بيجيب Department، إلا لما تطلبها.
  - مميزاته: تقليل الـ data المجلوبة.
  - عيوبه: ممكن يعمل **N+1 problem** (query لكل صف).
- **Eager (Include)**:
  - context.Employees.Include(e => e.Department).
  - يجيب كل حاجة في Query واحدة.
  - مميزاته: أسرع لو محتاج البيانات كلها.
  - عيوبه: ممكن يجيب داتا زيادة مش محتاجها.

3️⃣ Relationships Mapping

- **One-to-Many**: Department → Employees

- builder.HasOne(e => e.Department)
- .WithMany(d => d.Employees)
- .HasForeignKey(e => e.DeptId);
- **One-to-One**: Employee → Address
- builder.HasOne(e => e.Address)
- .WithOne(a => a.Employee)
- .HasForeignKey<Address>(a => a.EmployeeId);

📌 EF يترجم العلاقات دي لـ FK constraints في DB.

---

4 Soft Delete

- بدلاً من الحذف:
- public bool IsDeleted { get; set; }
- EF:
- modelBuilder.Entity<Employee>()
- .HasQueryFilter(e => !e.IsDeleted);
- مميزات:
  - الاحتفاظ بالتاريخ والـ logs.
  - أمان لو حصل delete بالغلط.
- عيوب:
  - حجم الداتا يكبر مع الوقت.

---

5 Delete Behaviors (Restrict, NoAction, Cascade)

- **Restrict**: (آمن) ما ينفعش تحذف الأب لو فيه أبناء.
- **NoAction**: يحذف الأب، ويسيب الأبناء يتيمه (ممكن Integrity problems).
- **Cascade**: (خطر لو ناسي علاقة) يحذف الأب والأبناء تلقائي.
- EF Core:

- builder.HasOne(e => e.Department)
- .WithMany()
- .OnDelete(DeleteBehavior.Restrict);

---

6 Change Tracking (AsTracking vs AsNoTracking)

- **AsTracking (default)**:
  - EF يراقب الـ entity.
  - لو عدّلت خاصية → EF يولّد Update.
- **AsNoTracking()**:
  - EF مش بيتابع التغييرات.
  - أسرع وأخف.
  - مناسب للتقارير أو Data readonly.
- مثال:
- var emps = context.Employees.AsNoTracking().ToList();

---

7 Round Trips to DB

- **Multiple Trips**:
- var emp = context.Employees.ToList();
- foreach (var e in emp)
- Console.WriteLine(e.Department.Name); // Query جديد لكل مرة
- **One Round Trip (أفضل)**:
- var emp = context.Employees.Include(e => e.Department).ToList();
- الهدف: تقليل عدد المرات اللي بتضرب فيها DB = أداء أفضل.