**First encounters**

Unix pipeline’a temel benzerligi, ilgili yöntemlerin her birinin (select, sortBy ve copyFrom) bir kayıt koleksiyonu üzerinde çalışması ve bir kayıt koleksiyonu döndürmesidir.Karmasik ve uzun işlemlerin lambda ile daha sade ve kisa sekilde yapilmasi.

**Defining Collection Pipeline**

Collection Pipeline'ı yazılımı nasıl modülerleştirebileceğimizin ve oluşturabileceğimizin bir modelidir.

Farkli modellerin birbirine bağlanarak icice baglanmis sorgu modeli .

**Exploring more pipelines and operations**

Bir datayi map,hashmap,groub by operatörleri kullanarak verileri istenilen veri dönüsüne getirmek.

**Alternatives**

Alternatif olarak döngüler kullanarak daha sade ve yalın kodlar yazılabilir.

**Nested Operator Expressions**

Bir diğer yolda ic ice döngü kümeleri yaratilmasidir. Küme işlemleriyle ilgili önemli nokta, koleksiyonların genellikle sıralı ve kopyalara izin veren listeler olmasıdır.

**Laziness**

## Bir listede arama yaparken ilk kac adet dönmesini gerektiğini söylendiği yapidir.Buyuk veri listelerinde performans acisindan önem gösteren bir kullanimdir.

**Parallelism**

## Bazi akislarin tekte serilik ama küme seklinde paralel olarak işleyebilmekteyiz .Bu bize performans kazanci sağlayabilir. seri ve paralel sürümleri arasındaki tek fark, "stream()" yerine "parallelStream()" kullanılarak ilk akışın oluşturulmasıdır.

**Immutability**

## Nesneler bir kez oluşturulduktan sonra içeriği değiştirilemeyen sınıflardır.

## Değişmez bir nesneyi değiştirmek isterseniz, onu klonlamanız ve oluştururken klonu değiştirmeniz gerekir.

## When to Use It

## Dil desteği sagladigi durumlarda ve karmasik datalarin tek cikisli döngülerinde kullanimi tercih edilir