Entity Framework

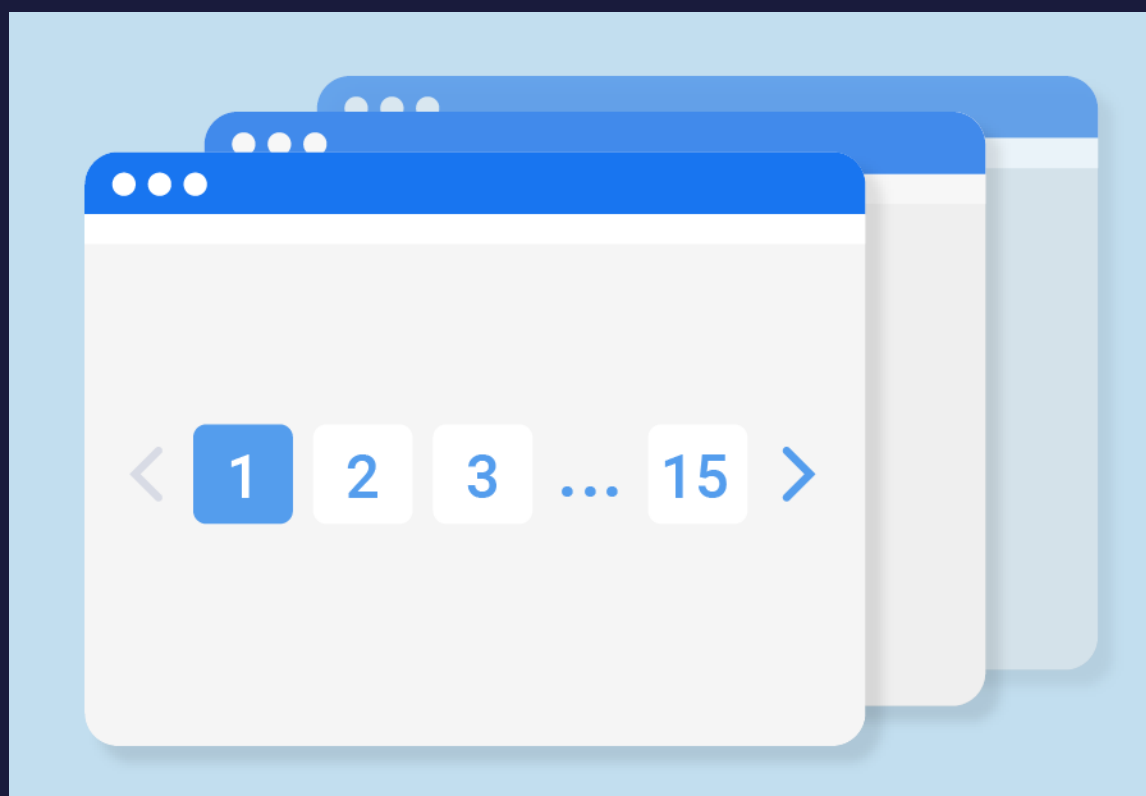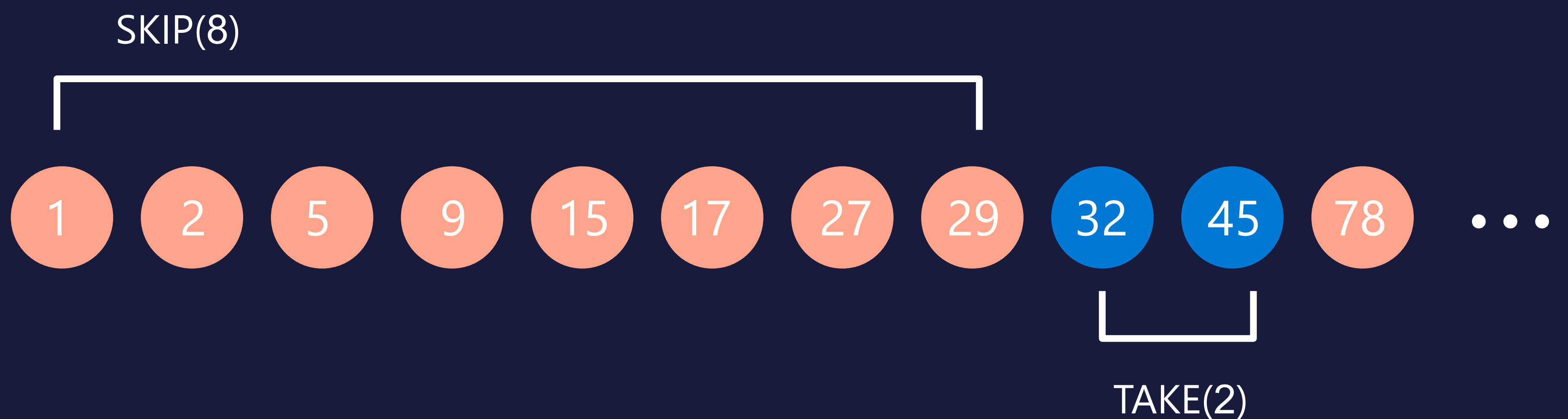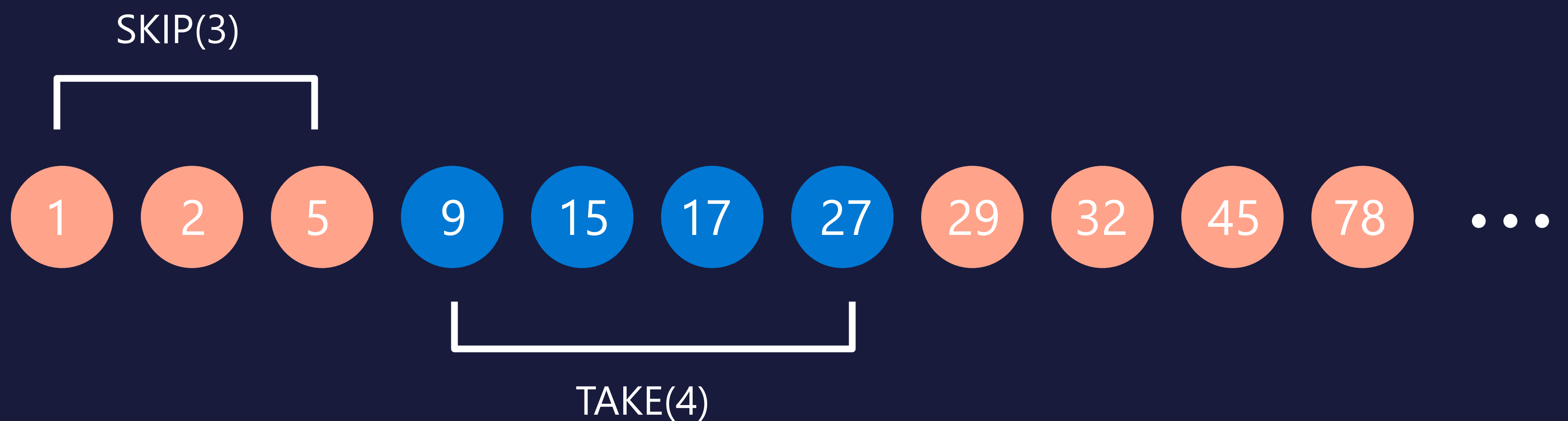💡 tips

Pagination
LINQ Extension

Pagination

It refers to retrieving results in pages, this is typically done for large resultsets.

Where a **user interface** is shown that allows the user to navigate to the next or previous page of the results.

In LINQ you can use Take and Skip methods for Implement pagination known as Paging.

SKIP(3)

| 1 | 2 | 5 | 9 | 15 | 17 | 27 | 29 | 32 | 45 | 78 | • • • |

TAKE(4)

SKIP(8)

| 1 | 2 | 5 | 9 | 15 | 17 | 27 | 29 | 32 | 45 | 78 | • • • |

TAKE(2)

Paging

Method

```
0 references | 0 changes | 0 authors, 0 changes
public static IQueryable<T> Page<T>(this IQueryable<T> data, int pageIndex, int pageSize)
            => data.Skip((pageIndex - 1) * pageSize)
                    .Take(pageSize);
```

Usage

```
var articlesOnPage = dbContex.Articles
                            .Page(pageIndex: 10, pageSize: 25);

// takes articles from 226 to 250
```

First-page

Method

```
0 references | 0 changes | 0 authors, 0 changes
public static IQueryable<T> FirstPage<T>(this IQueryable<T> data, int pageSize)
                => data.Take(pageSize);
```

Usage

```
var articlesOnFirstPage = dbContex.Articles
                                   .FirstPage(pageSize: 25);


// takes articles from 1 to 25
```

Last-page

Method

```
0 references | 0 changes | 0 authors, 0 changes
public static IQueryable<T> LastPage<T>(this IQueryable<T> data, int pageSize)
             => data.Skip(((data.Count() / pageSize) - 1) * pageSize)
                     .Take(pageSize);
```

Usage

```
var articlesOnLastPage = dbContex.Articles
                                  .LastPage(pageSize: 25);

// takes articles from 476 to 500 if you have 500 items
```

Page-Count

Method

```
1 reference | 0 changes | 0 authors, 0 changes
public static int CountOfPages<T>(this IQueryable<T> data, int pageSize)
{
    var total = data.Count();
    return (total / pageSize) + ((total % pageSize) > 0 ? 1 : 0);
}
```

Usage

```
int pageCount = dbContex.Articles
                        .CountOfPages(pageSize: 25);

// it's good when you want generate UI pager
```

Notice

Before Paging you must applying your filters and ordering by.

```csharp
var articlesOnPage = dbContex.Articles
                        .Where(art => art.AuthorId == authorId)
                        .OrderBy(art => art.Date)
                        .ThenBy(art => art.Title)
                        .Page(pageIndex: 10, pageSize: 25);
```

Do you want more post like this?

- Please follow me
- Click on the notification bell on my profile

@thisisnabi