Common operations of database query languages.

aka `for` expression with yield:`map`, `flatMap`, `filter`.

Without: `foreach`, `filter`.
aka `for` loop.

Creating a new collection from an old one.

```
if expr
```

`expr` is an expression resulting in a Boolean.

Filter drops from iteration the values for which `expr` returns `False`.

```
pat <- expr
```

`pat` is matched one-by-one against elements of `expr`, typically a List, but no `MatchError` is thrown; the element is simply discarded.

If `pat` is just a variable, the result will be simple iteration.

Before type checking. The only result of expansion must type check.

`map`, `flatMap`, `filter`, and `foreach` don't need and particular signature.

```
for (seq) yield expr
```

`seq` is a sequence of generators, definitions, and filters, with semicolons between elements.

`seq` starts with a generator.

```
pat = expr
```

This binds pattern `pat` to the value of `expr`.

The most common case is defining a simple variable x.

Note there is not necessarily a val. Simple variable binding will be the same as `val x = expr`.

If your type defines just `Map` it allows expressions with a single generator.

If it defines `flatMap + map` it allows expressions with multiple generators.

If it defines `foreach` it allows `for` loops (both with single and multiple generators).

If it defines `filter` it allows for filter expressions.