FunSQL

A library for compositional construction of SQL queries

https://github.com/MechanicalRabbit/FunSQL.jl

Clark C. Evans,
Kyrylo Simonov

JuliaCon 2021

Find all patients born in or after 1970.

```
SELECT p.person_id
FROM person p
WHERE p.year_of_birth >= 1970
```

```julia
function find_patients(conn)
    sql = """
    SELECT p.person_id
    FROM person p
    WHERE p.year_of_birth >= 1970
    """

    DBInterface.execute(conn, sql)
end
```

Find all patients born between [1970 ▲▼] *and* [▲▼]

```julia
function find_patients(conn; start_year = nothing, end_year = nothing)
    sql = """
    SELECT p.person_id
    FROM person p
    """

    predicates = String[]
    if start_year !== nothing
        push!(predicates, "p.year_of_birth >= $start_year")
    end
    if end_year !== nothing
        push!(predicates, "p.year_of_birth <= $end_year")
    end
    if !isempty(predicates)
        sql *=  "\nWHERE " * join(predicates, " AND ")
    end
    DBInterface.execute(conn, sql)
end
```

Find all patients born between `1970` ⬍ ── person.year_of_birth

and ⬍ ── location.state

living in `IL` ▾

with conditions `Hypertension` `Arrhythmia` `_____` ── condition_occurrence.condition_concept_id

who visited a doctor `within last 12 months` ▾ ── visit_occurrence.visit_start_date

```julia
function find_patients(conn; start_year = nothing,
                             end_year = nothing,
                             state = nothing,
                             conditions = [],
                             latest_visit = nothing)

    sql = ???

    DBInterface.execute(conn, sql)
end
```

A fragment of OMOP CDM
https://github.com/OHDSI/CommonDataModel

```julia
using FunSQL: SQLTable

const person =
    SQLTable(name = :person,
             columns = [:person_id, :year_of_birth, :location_id])

const location =
    SQLTable(name = :location,
             columns = [:location_id, :city, :state, :zip])

const visit_occurrence =
    SQLTable(name = :visit_occurrence,
             columns = [:visit_occurrence_id, :person_id, :visit_concept_id,
                        :visit_start_date, :visit_end_date])
```

Find all patients born in or after 1970.

```
using FunSQL: From, Get, Select, Where, render
```

FROM person p

⬇

```
q = From(person)
```

⬇

FROM person p
WHERE p.year_of_birth >= 1970

⬇

```
q = From(person) |>
    Where(Get.year_of_birth .>= 1970)
```

⬇

SELECT p.person_id
FROM person p
WHERE p.year_of_birth >= 1970

```
q = From(person) |>
    Where(Get.year_of_birth .>= 1970) |>
    Select(Get.person_id)
```

```
sql = render(q, dialect = :postgresql)
```

```
q₁ = From(person)
q₂ = q₁ |> Where(q₁.year_of_birth .>= 1970)
q₃ = q₂ |> Select(q₂.person_id)
```
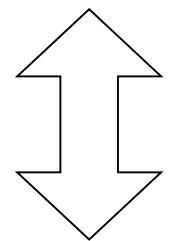
bound references

person	
PK	person_id
	year_of_birth
FK	location_id

```
q = From(person) |>
    Where(Get.year_of_birth .>= 1970) |>
    Select(Get.person_id)
```
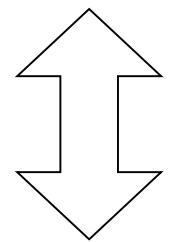
unbound references

```
BornInOrAfter(Y) = Get.year_of_birth .>= Y

q = From(person) |>
    Where(BornInOrAfter(1970)) |>
    Select(Get.person_id)
```

```
                                        using FunSQL: Fun
                              "Fun" notation

SELECT p.person_id                      Fun.">="(Get.year_of_birth, 1970)
FROM person p                                       or
WHERE p.year_of_birth >= 1970           Get.year_of_birth .>= 1970

                              broadcasting


WHERE p.year_of_birth >= 1970 AND       Fun.and(Get.year_of_birth .>= 1970,
      p.year_of_birth <= 2000                   Get.year_of_birth .<= 2000)



WHERE p.year_of_birth                   Fun.between(Get.year_of_birth, 1970, 2000)
      BETWEEN 1970 AND 2000
```

Show patients with their state of residence.

```
FROM person p
```

⬇

```
FROM person p
JOIN location l
  ON (p.location_id = l.location_id)
```
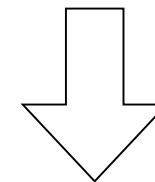
⬇
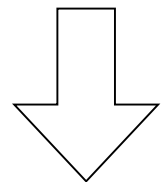
```
SELECT p.person_id, l.state
FROM person p
JOIN location l
  ON (p.location_id = l.location_id)
```

```
using FunSQL: Join
```

```
q = From(person)
```

⬇

```
q = From(person) |>
    Join(:location => location,
         Get.location_id .==
         Get.location.location_id)
```

⬇
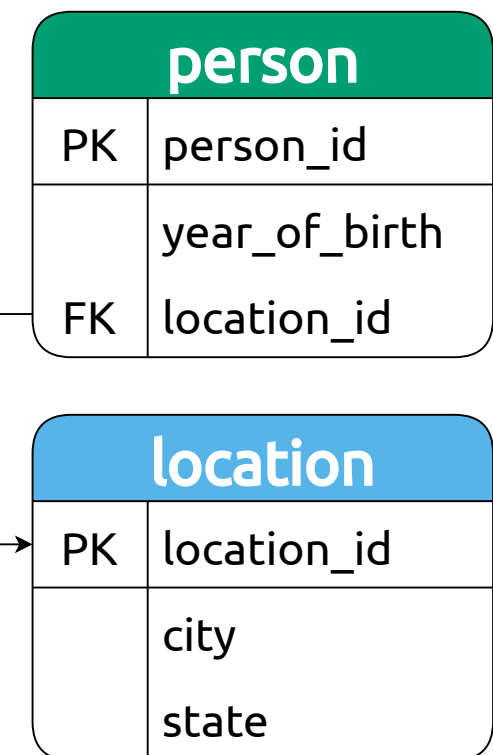
```
q = From(person) |>
    Join(:location => location,
         Get.location_id .==
         Get.location.location_id) |>
    Select(Get.person_id, Get.location.state)
```

person

PK	person_id
	year_of_birth
FK	location_id

location

PK	location_id
	city
	state

Find patients

- *born in or after 1970*
- *living in Illinois*

person

PK	person_id
	year_of_birth
FK	location_id

location

PK	location_id
	city
	state

```
q_p = From(person) |>
    Where(Get.year_of_birth .>= 1970)
```

```
q_l = From(location) |>
    Where(Get.state .== "IL") |>
    As(:location)
```
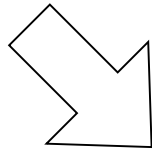
```
q_p |> Join(q_l,
            Get.location_id .==
            Get.location.location_id)
```