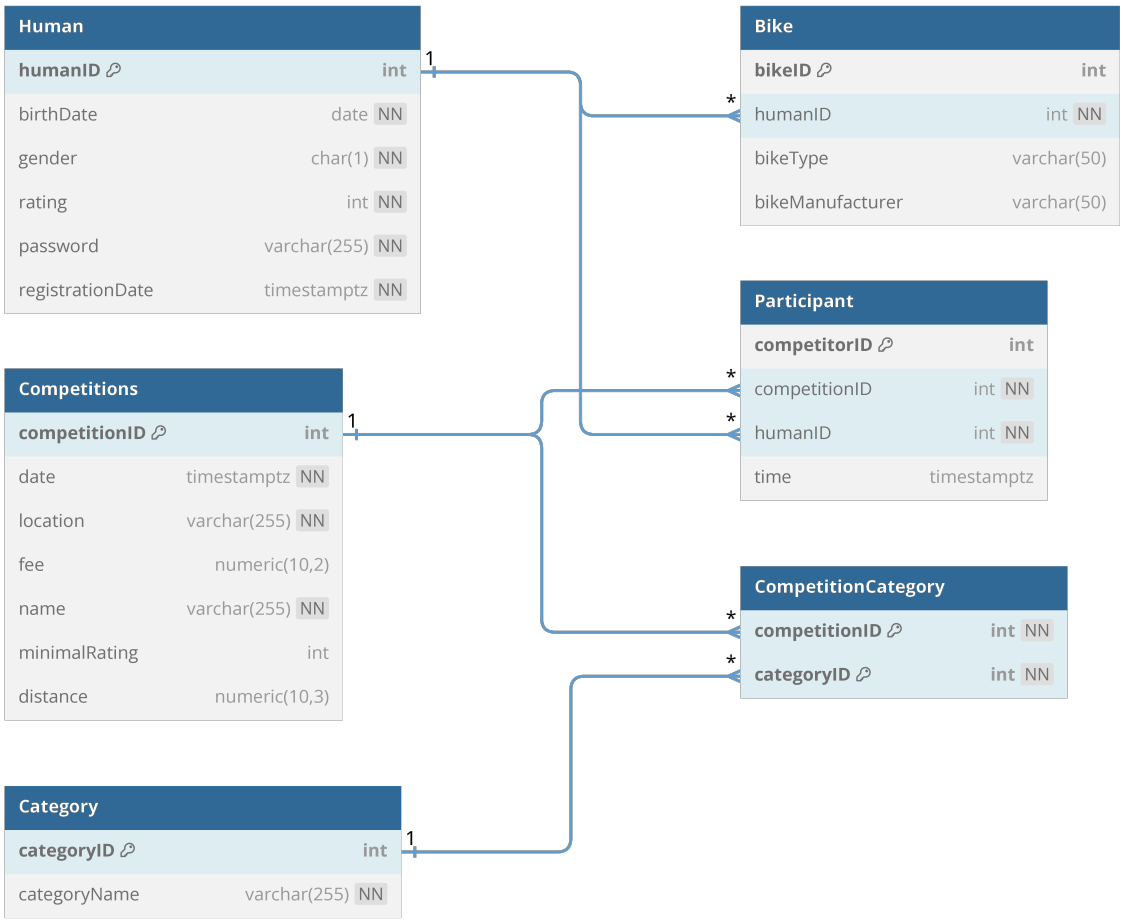
Biking Competition Database

Overview

This database system is designed to manage biking competitions, including participant registration, bicycle information, competition details and results tracking. The system supports user authentication, competition categorization and performance analytics.

Database Structure

Entity Relationship Diagram



View database relations online: `https://dbdocs.io/frybaylt/Biking-Competition-Database?view=relationships`

Tables Description

1. Human

Stores information about registered users.

- **Primary Key:** `humanID` (auto-generated)
- **Key Features:**
  - Birth date validation (cannot be future date)
  - Gender input (M/F/O/N)
  - Rating system (0–3000 range, default 1000)
  - Password security (minimum 6 characters, must contain numbers and letters)
  - Registration timestamp tracking

2. Bike

Contains bicycle information owned by users.

- **Primary Key:** `bikeID` (auto-generated)
- **Foreign Key:** `humanID` → Human
- **Features:**
  - Bicycle type and manufacturer tracking
  - One-to-many relationship with Human

3. Competitions

Main competition information.

- **Primary Key:** `competitionID` (auto-generated)
- **Key Features:**
  - Competition scheduling with timestamp
  - Location and distance tracking
  - Entry fee management
  - Minimum rating requirements
  - Unique competition names

4. Participant

Junction table linking participants to competitions with results.

- **Primary Key:** `competitorID` (auto-generated)
- **Foreign Keys:** `competitionID` → Competitions, `humanID` → Human
- **Features:** Completion time tracking; many-to-many relationship

5. Category

Competition categories for classification.

- **Primary Key:** `categoryID` (auto-generated)
- **Features:** Category name storage; supports multiple categories per competition

6. CompetitionCategory

Many-to-many relationship between competitions and categories.

- **Composite Primary Key:** `competitionID`, `categoryID`
- **Purpose:** Allows competitions to belong to multiple categories

Database Features

Views

- `HumanAge`: calculates current age of all persons
- `ParticipantCount`: shows participant count for each competition

Indexes

- `competitionNameIndex`: ensures unique competition names
- `humanBikesIndex`: optimizes bicycle queries by person

Functions

- `registerUser()`: handles user registration
- `registerBike()`: manages bicycle registration
- `registerCompetition()`: creates new competitions

PostgreSQL Implementation

```
CREATE TABLE Human (
    humanID int GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    birthDate date NOT NULL CHECK (birthDate <= CURRENT_DATE),
    gender char(1) NOT NULL CHECK (gender IN ('M', 'F', 'O', 'N')),
    rating int NOT NULL DEFAULT 1000 CHECK (rating BETWEEN 0 AND 3000),
    PASSWORD VARCHAR(255) NOT NULL,
    registrationDate timestamptz NOT NULL DEFAULT CURRENT_DATE,
    CONSTRAINT passwordValidation CHECK (length(PASSWORD) >= 6 AND
       PASSWORD ~ '[0-9]' AND PASSWORD ~ '[a-zA-Z]')
);

CREATE TABLE Bike (
    bikeID int GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    humanID int NOT NULL,
    bikeType varchar(50),
    bikeManufacturer varchar(50),
    FOREIGN KEY (humanID) REFERENCES Human (humanID) ON DELETE CASCADE ON
       UPDATE CASCADE
);

CREATE TABLE Competitions (
    competitionID int GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    date timestamptz NOT NULL,
    location varchar(255) NOT NULL,
    fee numeric(10, 2) DEFAULT '0.00',
    name varchar(255) NOT NULL,
    minimalRating int CHECK (minimalRating >= 0),
    distance numeric(10, 3)
);

CREATE TABLE Participant (
    competitorID int GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    competitionID int NOT NULL,
    humanID int NOT NULL,
    time timestamptz,
    FOREIGN KEY (competitionID) REFERENCES Competitions (competitionID) ON
        DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (humanID) REFERENCES Human (humanID) ON DELETE CASCADE ON
       UPDATE CASCADE
);

CREATE TABLE Category (
    categoryID int GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    categoryName varchar(255) NOT NULL
);

CREATE TABLE CompetitionCategory (
    competitionID int NOT NULL,
    categoryID int NOT NULL,
    PRIMARY KEY (competitionID, categoryID),
```

```
    FOREIGN KEY (competitionID) REFERENCES Competitions (competitionID) ON
        DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (categoryID) REFERENCES Category (categoryID) ON DELETE
       CASCADE ON UPDATE CASCADE
);

CREATE UNIQUE INDEX competitionNameIndex ON Competitions (name);

CREATE INDEX humanBikesIndex ON Bike (humanID);

CREATE VIEW HumanAge AS
SELECT
    humanID,
    DATE_PART('year', AGE(birthDate)) AS age
FROM
    Human;

CREATE VIEW ParticipantCount AS
SELECT
    Competitions.competitionID,
    Competitions.name AS name,
    COUNT(*) AS participantCount
FROM
    Participant
    INNER JOIN Competitions ON Participant.competitionID = Competitions.
       competitionID
GROUP BY
    Competitions.competitionID,
    Competitions.name;

CREATE FUNCTION registerUser (birthDate date, gender char(1), rating int,
   PASSWORD VARCHAR(255), registrationDate timestamptz)
    RETURNS VOID
    AS $$
BEGIN
    INSERT INTO Human (birthDate, gender, rating, password,
       registrationDate)
        VALUES (birthDate, gender, rating, password, registrationDate);
END;
$$
LANGUAGE plpgsql;

CREATE FUNCTION registerBike (humanID int, bikeType varchar(50),
   bikeManufacturer varchar(50))
    RETURNS VOID
    AS $$
BEGIN
    INSERT INTO Bike (humanID, bikeType, bikeManufacturer)
        VALUES (humanID, bikeType, bikeManufacturer);
END;
$$
LANGUAGE plpgsql;
```

```
CREATE OR REPLACE FUNCTION registerCompetition (date timestamptz, location
    varchar(255), fee numeric(10, 2), name varchar(255), minimalRating int
   , distance numeric(10, 3))
    RETURNS VOID
    AS $$
BEGIN
    INSERT INTO Competitions (date, location, fee, name, minimalRating,
       distance)
        VALUES (date, location, fee, name, minimalRating, distance);
END;
$$
LANGUAGE plpgsql;
```

Sample Data

```
INSERT INTO Human (birthDate, gender, rating, password, registrationDate)
    VALUES ('1980-01-15', 'M', 1200, 'Slaptazodis123!', NOW()),
    ('1995-03-21', 'F', 1500, '$laptas123!', NOW()),
    ('2000-07-10', 'N', 800, 'ManoSlaptasZodis123', NOW());

INSERT INTO Bike (humanID, bikeType, bikeManufacturer)
    VALUES (1, 'City', 'Trek'),
    (2, 'MTB', 'Giant'),
    (1, 'Gravel', 'Scott');

INSERT INTO Competitions (date, location, fee, name, minimalRating,
   distance)
    VALUES ('2024-06-15 10:00:00', 'Panev  io senvag ', 15, 'Dvira i
        maratonas 2024', 800, 50),
    ('2024-07-20 09:00:00', 'Vilniaus miesto centras', 10, 'Naktinis
       Vilniaus kriteriumas', 1000, 20);

INSERT INTO Category (categoryName)
    VALUES ('Friendly'),
    ('Rated'),
    ('TimeTrial');

INSERT INTO CompetitionCategory (competitionID, categoryID)
    VALUES (1, 1),
    (1, 3),
    (2, 2);

INSERT INTO Participant (competitionID, humanID, time)
    VALUES (1, 1, '2024-06-15 12:30:00'),
    (1, 2, '2024-06-15 12:30:15'),
    (2, 2, '2024-07-20 10:15:00');
```