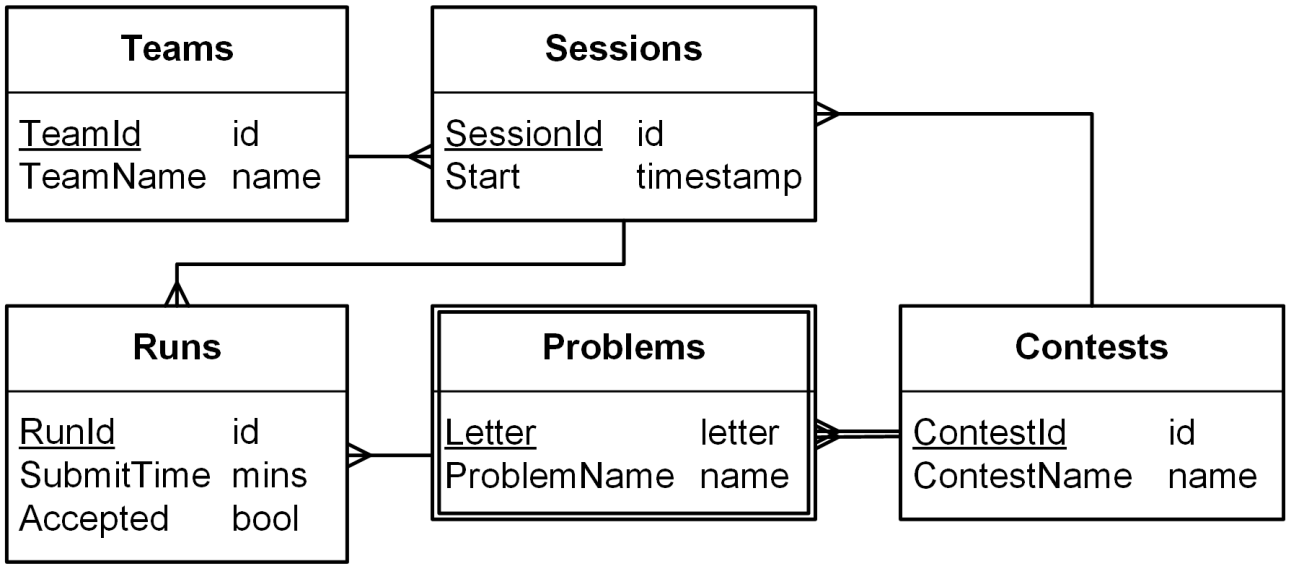
Структура базы данных



В таблице `Runs`: `SubmitTime` – целое число минут прошедших с начала соревнования, `Accepted` – `1`, если зачтено, иначе `0`.

Примеры исходных данных можно найти в тестовом полигоне:
\https://www.kgeorgiy.info/courses/dbms/slides/misc-En2okepgeegyitPotod0/relations.html

Реализуйте запросы к базе данных «Соревнования по программированию»

1. Реляционная алгебра

Запишите следующие запросы в терминах реляционной алгебры и на языке SQL

1. Идентификаторы команд, участвующих в соревновании
   (`TeamId` по `:ContestId`).

2. Названия команд, участвующих в соревновании
   (`TeamName` по `:ContestId`).

3. Информация о подходах по задаче в соревновании
   (`RunId`, `TeamId`, `SubmitTime`, `Accepted` по `:Letter`, `:ContestId`).

4. Неудачные подходы в соревновании
   (`RunId`, `SessionId`, `Letter`, `SubmitTime` по `:ContestId`).

5. Задачи, не решённые ни одной командной
   (`ProblemName`).

6. Задачи без подходов
   (`ProblemName`).

7. Команды, не сделавшие ни одного подхода хотя бы в одном соревновании
   (`TeamName`).

8. Задачи, по которым есть подходы всех сессиий в соревновании
   (`ContestId`, `Letter`).

9. Команды, решившие все задачи хотя бы в одном соревновании
   (`TeamName`).

10. Задачи, в которых ошиблись все команды, участвовавшие в соревновании
    (`ContestId`, `Letter`).

2. Реляционное исчисление

Запишите следующие запросы на языках Datalog и SQL

1. Идентификаторы команд, ошибшихся в задаче
   (`TeamId` по `:ContestId`, `:Letter`).

2. Названия команд, не решивших задачу
   (`TeamName` по `:ContestId`, `:Letter`).

3. Команды, ошибившиеся хотя бы в одной задачи в соревнования
   (`TeamId` по `:ContestId`).

4. Задачи, решённые всеми сессиями, участвовашими в соревновании
   (`ContestId`, `Letter`).

5. Команды, решившие все задачи, не решённые заданной командой
   (`TeamId` по `:TeamId`).

6. Задачи, которые не решила ни одна сессия, участвовавшая в соревновании
   (`ProblemName`).

3. Изменяющие запросы

Запишите изменяющие запросы на языке SQL

1. Удалить все подходы команды
   (`TeamId`).

2. Удалить все подходы соревнования
   (`ContestName`).

3. Для каждой команды, участвовавшей в соревновании добавить новую сессию с текущим временем начала (`current_timestamp`)
   (`ContestId`).

4. Сделать последний ошибочный подход в каждой сессии успешным.

5. Сделать первый подход по каждой задаче в каждой сессии ошибочным.

6. Для каждой сессии, пытавшейся, но не решившей задачу, добавить успешный подход через минуту после последнего подхода по этой задаче
   (`ContestId`).

7. Для каждой сессии сделать успешный подход по задаче с нулевым временем. Если задача уже была решена, то изменить время успешных подходов. [Не проверяется на `SQLite`.]
   (`ContestId`, `Letter`).

4. Агрегирующие запросы

Запишите агрегирующие запросы на языке SQL

1. Для каждой сессии число задач, по которым были подходы
   (`SessionId`, `Opened`).

2. Для каждой команды число задач, по которым были подходы
   (`TeamId`, `Opened`).

3. Задачи соревнования, которые решило минимальное число сессий
   (`Letter` по `:ContestId`).

4. Для каждого соревнования, задачи, которые решило максимальное число сессий
   (`ContestId`, `Letter`).

5. Месяцы, в которые создано максимальное число сессий, решивших хотя бы одну задачу, в формате `mm-yyyy`. [Не проверяется на `SQLite`.]
   (`MonthStr`).

6. Посчитать число задач, решённых командами. Команды должны быть упорядочены по числу решешенных задач; при равенстве результатов — по убыванию времени начала.
   (`TeamName`, `Solved` по `:ContestId`).

7. Посчитать штрафное время, полученное командами. Команды должны быть упорядочены по штрафному времени; при равенстве результатов — по убыванию времени начала.
   (`TeamName`, `Solved`, `Penalty` по `:ContestId`).

8. Построить по соревнованию колонки «решено задач» и «штрафное время» Порядок команд должен быть правильным; при равенстве результатов — по убыванию времени начала.
   (`TeamName`, `Solved`, `Penalty` по `:ContestId`).