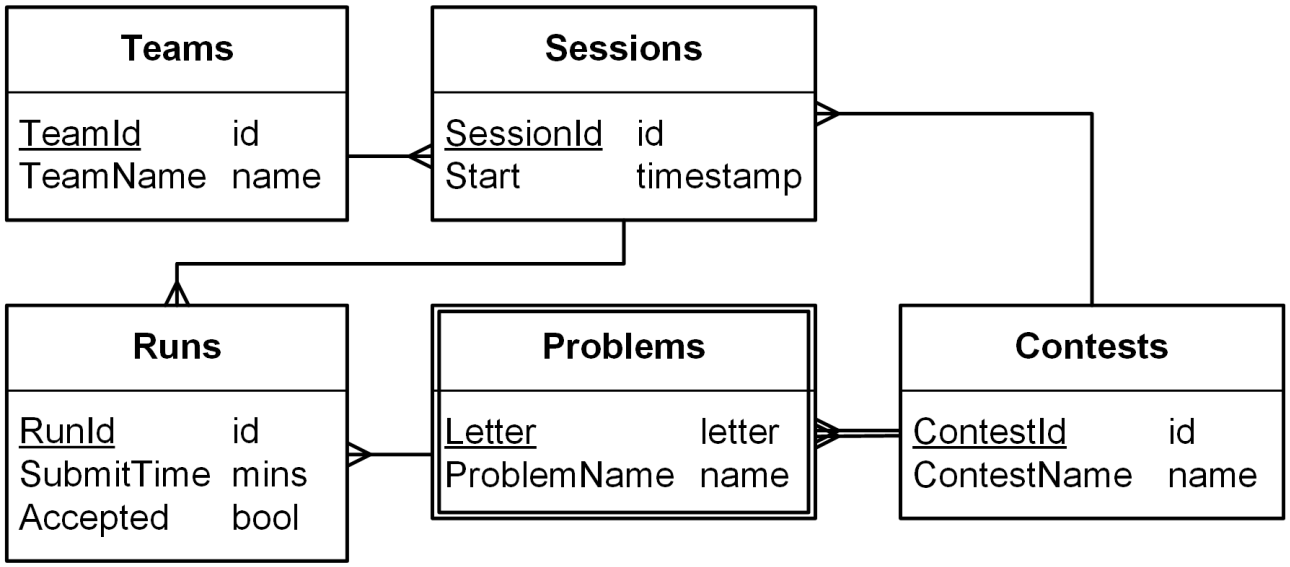
Структура базы данных



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/relations.html

1. Реляционная алгебра

Запишите следующие запросы в терминах реляционной алгебры и на языке SQL.

1. Информация о подходах команды `:TeamId` в соревновании `:ContestId`.
   Вернуть `RunId`, `SessionId`, `Letter`, `SubmitTime`, `Accepted`.

2. Успешные подходы в соревновании `:ContestId`.
   Вернуть `RunId`, `SessionId`, `Letter`, `SubmitTime`.

3. Команды, не решившие ни одной задачи ни в одном соревновании.
   Вернуть `TeamName`.

4. Команды, не решившие ни одной задачи хотя бы в одном соревновании.
   Вернуть `TeamName`.

5. Команды, не решившие ни одной задачи хотя бы в одном соревновании, в котором они участвовали.
   Вернуть `TeamName`.

6. Сессии, имеющие подходы по всем задачам в соревновании.
   Вернуть `SessionId`.

7. Сессии, в которых решены все задачи в соревновании
   Вернуть `SessionId`.

8. Команды, решившие все задачи хотя бы в одном соревновании (возможно, в разных сессиях)
   Вернуть `TeamName`.

9. Задачи, которые решили все команды участвовавшие в соревновании
   Вернуть `ContestId`, `Letter`.

2. Реляционное исчисление

Запишите следующие запросы на языках Datalog и SQL.

1. Команды, решившие задачу по `:ContestId` и `:Letter`.
   Вернуть `TeamId`.
2. Команды, решившие задачу по `:ContestId` и `:Letter`.
   Вернуть `TeamName`.
3. Команды, решившие хотя бы одну задачу в соревновании `:ContestId`.
   Вернуть `TeamId`.
4. Задачи, которые не решила ни одна команда.
   Вернуть `ContestId`, `Letter`.
5. Команды, решившие все задачи, решённые командой `:TeamId` (возможно, в разных сессиях).
   Вернуть `TeamId`.
6. Задачи, которые решили все команды участвовавшие в соревновании.
   Вернуть `ProblemName`.

3. Изменения

Запишите следующие запросы на языке SQL.

1. Удалить все попытки по `:ContestId`.
2. Удалить все попытки по `:TeamName`.
3. Для каждой команды, не участвовавшей в соревновании `:ContestId` добавить сессию с текущим временем начала (`current_timestamp`).
4. Сделать последний подход в каждой сессии успешным (если в сессии есть хотя бы один подход).
5. Сделать последний подход по каждой задаче в каждой сессии успешным (если есть подходы).
6. Для каждой команды сделать сессию для соревнования `:ContestId` с текущим временем начала. Если сессии уже существовали, то изменить их время начала. Не проверяется на SQLite.

4. Агрегирующие запросы

Запишите следующие запросы на языке SQL.

1. Число задач, решенных в каждой сессии. Вернуть `SessionId`, `Solved`.
2. Число различных задач, решенных командой. Вернуть `SessionId`, `Solved`.
3. Задачи, которые решили максимальное число команд по `:ContestId`
   Вернуть `Letter`.
4. Для каждого соревнования: задачи, которые решили максимальное число команд
   Вернуть `ContestId`, `Letter`.
5. Месяцы, в которые создано максимальное число сессий.
   Вернуть `MonthStr` в формате *`месяц-год`*. Не проверяется на SQLite.
6. Для каждого соревнования найти команду, совершившую последнюю удачную попытку в этом соревновании.
   Вернуть `ContestId`, `TeamName`, `SubmitTime`.
7. Построить колонки «решено задач» и «штрафное время» по `:ContestId` (с правильным порядком; при равенстве результатов — по возрастанию `SessionId`).
   Вернуть `TeamName`, `Solved`, `Penalty`.