1 Концептуальная модель

Тема: Спортивный клуб.
Сущности: тренер, спортзал, вид спорта, спортсмен, расписание занятий, соревнование.

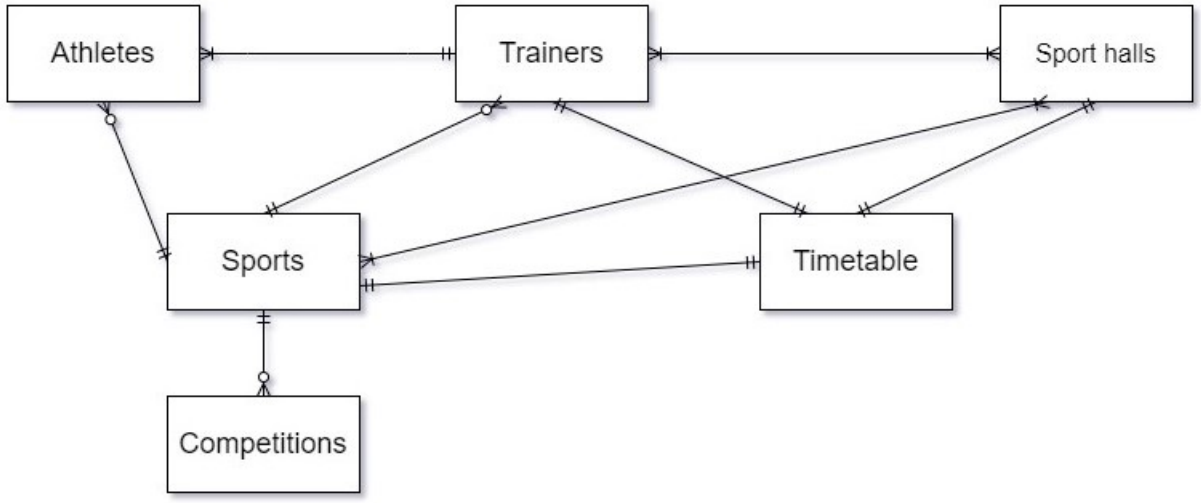

Рис. 1: концептуальная модель.

2 Логическая модель

Я выбрала для базы данных *Третью нормальную форму (3НФ)*, значения во всех столбце не зависят ни от одного ключа, каждое не ключевое поле полностью зависит от первичного ключа.
Таблица Former Athletes является версионной, четвертого типа.

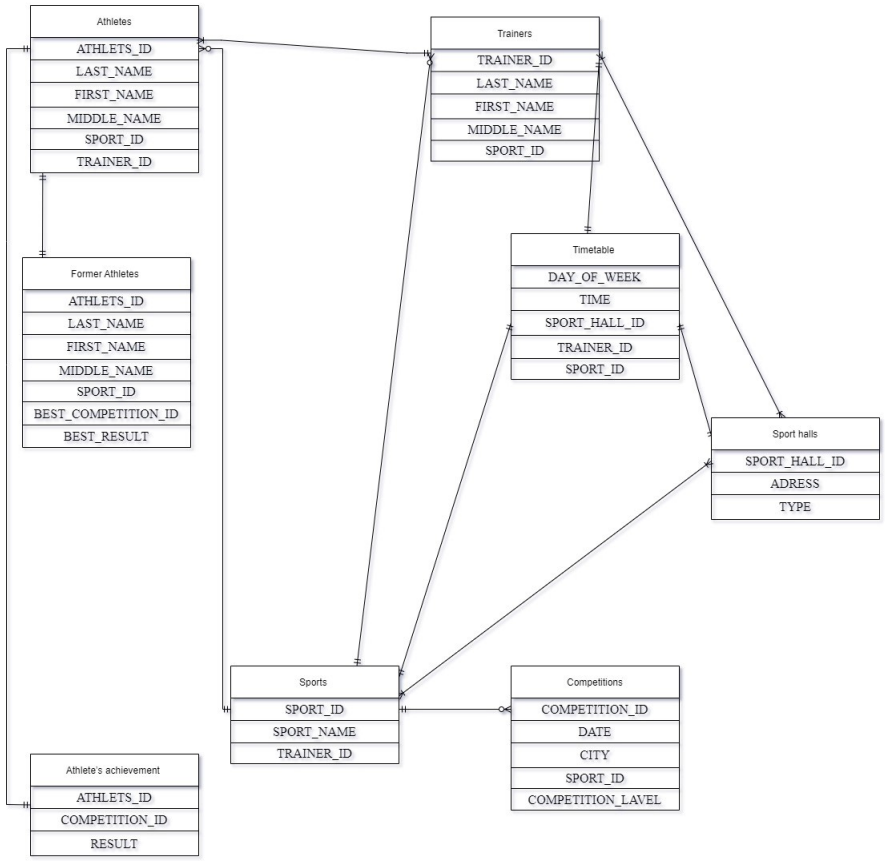

Рис. 2: Логическая модель.

3 Физическая модель

3.1 Физическая модель

Trainers Тренеры			
Название	Описание	Тип данных	Ограничение
TRAINER_ID	Идентификатор тренера	INTEGER	NOT NULL
LAST_NAME	Фамилия	VARCHAR(200)	
FIRST_NAME	Имя	VARCHAR(200)	
MIDDLE_NAME	Отчество	VARCHAR(200)	
SPORT_ID	Идентификатор вида спорта	INTEGER	NOT NULL

Sports Виды спорта			
Название	Описание	Тип данных	Ограничение
SPORT_ID	Идентификатор вида спорта	INTEGER	NOT NULL
SPORT_NAME	Название вида спорта	VARCHAR(200)	
TRAINER_ID	Идентификатор ответственного	INTEGER	

Timetable Расписание тренировок			
Название	Описание	Тип данных	Ограничение
DAY_OF_WEEK	День недели	VARCHAR(200)	
TIME	Время	TIME	
SPORT_HALL_ID	Идентификатор спортзала	INTEGER	
TRAINER_ID	Идентификатор тренера	INTEGER	
SPORT_ID	Идентификатор вида спорта	INTEGER	

Athlete's achievement Достижения спортсменов			
Название	Описание	Тип данных	Ограничение
ATHLETS_ID	Идентификатор спортсмена	INTEGER	NOT NULL
COMPETITION_ID	Идентификатор соревнований	INTEGER	
RESULT	Результат	TEXT	

Competitions Соревнования			
Название	Описание	Тип данных	Ограничение
COMPETITION_ID	Идентификатор соревнования	INTEGER	NOT NULL
DATE	Дата	DATE	
CITY	Город	VARCHAR(200)	
SPORT_ID	Идентификатор вида спорта	INTEGER	
COMPETITION_-LAVEL	Уровень соревнования	TEXT	

Athletes Спортсмены			
Название	Описание	Тип данных	Ограничение
ATHLETS_ID	Идентификатор спортсмена	INTEGER	NOT NULL
LAST_NAME	Фамилия	VARCHAR(200)	
FIRST_NAME	Имя	VARCHAR(200)	
MIDDLE_NAME	Отчество	VARCHAR(200)	
SPORT_ID	Идентификатор вида спорта	INTEGER	
TRAINER_ID	Идентификатор тренера	INTEGER	

Sport halls Спортзалы			
Название	Описание	Тип данных	Ограничение
SPORT_HALL_ID	Идентификатор спортзала	INTEGER	NOT NULL
ADRESS	Адрес	TEXT	
TYPE	Тип зала	VARCHAR(200)	

Former Athletes Бывшие спортсмены			
Название	Описание	Тип данных	Ограничение
ATHLETS_ID	Идентификатор спортсмена	INTEGER	NOT NULL
LAST_NAME	Фамилия	VARCHAR(200)	
FIRST_NAME	Имя	VARCHAR(200)	
MIDDLE_NAME	Отчество	VARCHAR(200)	
SPORT_ID	Идентификатор вида спорта	INTEGER	
TRAINER_ID	Идентификатор тренера	INTEGER	
BEST_-COMPETITION_ID	Идентификатор лучшего соревнования	INTEGER	
BEST_RESULT	Лучший результат	TEXT	

3.2 DDL скрипты

```
DROP SCHEMA IF EXISTS project CASCADE;
CREATE SCHEMA project;

DROP TABLE IF EXISTS project.Trainers;
CREATE TABLE project.Trainers (
 TRAINER_ID              INTEGER        NOT NULL PRIMARY KEY,
 LAST_NAME               VARCHAR(200),
 FIRST_NAME              VARCHAR(200),
 MIDDLE_NAME             VARCHAR(200)
);
DROP TABLE IF EXISTS project.Sports;
CREATE TABLE project.Sports (
 SPORT_ID                INTEGER        NOT NULL PRIMARY KEY,
 SPORT_NAME              VARCHAR(200)
);
DROP TABLE IF EXISTS project.Timetable;
CREATE TABLE project.Timetable (
 DAY_OF_WEEK             VARCHAR(200),
 TIME                    TIME,
 SPORT_HALL_ID           INTEGER
);
DROP TABLE IF EXISTS project.Athlete's_achievement;
```

```
CREATE TABLE project.Athlete's_achievement (
 ATHLETS_ID                INTEGER        NOT NULL PRIMARY KEY,
 RESULT                    TEXT
);
DROP TABLE IF EXISTS project.Competitions;
CREATE TABLE project.Competitions (
 COMPETITION_ID            INTEGER        NOT NULL PRIMARY KEY,
 DATE                      DATE,
 CITY                      VARCHAR(200),
 COMPETITION_LAVEL         TEXT
);
DROP TABLE IF EXISTS project.Sport_halls;
CREATE TABLE project.Sport_halls (
 SPORT_HALL_ID             INTEGER        NOT NULL PRIMARY KEY,
 ADRESS                    TEXT,
 TYPE                      VARCHAR(200)
);
DROP TABLE IF EXISTS project.Athletes;
CREATE TABLE project.Athletes (
 ATHLETS_ID                INTEGER        NOT NULL PRIMARY KEY,
 LAST_NAME                 VARCHAR(200),
 FIRST_NAME                VARCHAR(200),
 MIDDLE_NAME               VARCHAR(200)
);
DROP TABLE IF EXISTS project.Former_Athletes;
CREATE TABLE project.Former_Athletes (
 ATHLETS_ID                INTEGER        NOT NULL PRIMARY KEY,
 LAST_NAME                 VARCHAR(200),
 FIRST_NAME                VARCHAR(200),
 MIDDLE_NAME               VARCHAR(200),
 BEST_COMPETITION_ID       INTEGER,
 BEST_RESULT               TEXT
);
ALTER TABLE project.Trainers ADD COLUMN SPORT_ID INTEGER
REFERENCES project.Sports(SPORT_ID);
ALTER TABLE project.Sports ADD COLUMN TRAINER_ID INTEGER
REFERENCES project.Trainers(TRAINER_ID);
ALTER TABLE project.Timetable ADD COLUMN TRAINER_ID INTEGER
REFERENCES project.Trainers(TRAINER_ID);
ALTER TABLE project.Timetable ADD COLUMN SPORT_ID INTEGER
REFERENCES project.Sports(SPORT_ID);
```

ALTER TABLE project.Athlete's_achievement ADD COLUMN COMPETITION_-
ID INTEGER REFERENCES project.Competitions(COMPETITION_ID);
ALTER TABLE project.Competitions ADD COLUMN SPORT_ID INTEGER
REFERENCES project.Sports(SPORT_ID);
ALTER TABLE project.Athletes ADD COLUMN SPORT_ID INTEGER
REFERENCES project.Sports(SPORT_ID);
ALTER TABLE project.Athletes ADD COLUMN TRAINER_ID INTEGER
REFERENCES project.Trainers(TRAINER_ID);
ALTER TABLE project.Former_Athletes ADD COLUMN SPORT_ID INTEGER
REFERENCES project.Sports(SPORT_ID);
ALTER TABLE project.Former_Athletes ADD COLUMN TRAINER_ID
INTEGER REFERENCES project.Trainers(TRAINER_ID);

3.3 Запросы insert,select,update,delete

INSERT INTO project.competitions VALUES (6, '13.01.13', 'Екатеренбург',
'Первенство спортивной школы №2' , 4);
INSERT INTO project.former_athletes VALUES (10, 'Белинская', 'Екате-
рина', 'Владимировна', 3, 'Приз зрительских симпатий', 7, 7);
INSERT INTO project.former_athletes VALUES (3, 'Ефимова', 'Елена',
'Михайловна', 10, 'Чемпион', 9, 9);
INSERT INTO project.Athlete's_achievement VALUES (1, 'III место', 2);

UPDATE project.sports SET trainer_id = 1 WHERE sport_id = 1;
UPDATE project.sports SET trainer_id = 2 WHERE sport_id = 2;
UPDATE project.sports SET trainer_id = 3 WHERE sport_id = 3;
UPDATE project.sports SET trainer_id = 4 WHERE sport_id = 4;
UPDATE project.sports SET trainer_id = 5 WHERE sport_id = 5;
UPDATE project.sports SET trainer_id = 6 WHERE sport_id = 6;
UPDATE project.sports SET trainer_id = 7 WHERE sport_id = 7;
UPDATE project.sports SET trainer_id = 8 WHERE sport_id = 8;
UPDATE project.sports SET trainer_id = 9 WHERE sport_id = 9;
UPDATE project.sports SET trainer_id = 10 WHERE sport_id = 10;

SELECT * FROM project.timetable;
SELECT * FROM project.competitions;
SELECT * FROM project.athletes;
SELECT * FROM project.sport_halls;
SELECT * FROM project.sports LEFT JOIN project.trainers ON project.sports.sport_-
id = project.trainers.sport_id;

SELECT * FROM project.former_athletes;
SELECT * FROM project.Athlete's_achievement;
SELECT * FROM project.trainers;
SELECT * FROM project.sports;

INSERT INTO project.trainers VALUES (11, 'Кирилл', 'Кириллович', 'Кирилов', 4);
INSERT INTO project.athletes VALUES (100500, 'Филатов', 'Киррил', 'Иванович', 4, 11);
INSERT INTO project.Athlete's_achievement VALUES (100500, NULL, 1);
INSERT INTO project.Athlete's_achievement VALUES (100500, NULL, 2);
INSERT INTO project.Athlete's_achievement VALUES (100500, NULL, 4);
INSERT INTO project.Athlete's_achievement VALUES (100500, NULL, 6);
INSERT INTO project.Athlete's_achievement VALUES (100500, NULL, 8);
INSERT INTO project.athletes VALUES (100501, 'Макаров', 'Глеб', 'Витальевич', 4, 11);
INSERT INTO project.Athlete's_achievement VALUES (100501, 'I место', 1);

DELETE FROM project.Athlete's_achievement WHERE ATHLETS_ID = 100500;
DELETE FROM project.athletes WHERE ATHLETS_ID = 100500;
DELETE FROM project.Athlete's_achievement WHERE ATHLETS_ID = 100501;
DELETE FROM project.athletes WHERE ATHLETS_ID = 100501;
DELETE FROM project.trainers WHERE trainer_ID = 11;

INSERT INTO project.former_athletes VALUES (100500, 'Филатов', 'Киррил', 'Иванович', NULL, NULL, 4, 11);
INSERT INTO project.former_athletes VALUES (100501, 'Макаров', 'Глеб', 'Витальевич', 1, 'I место', 4, 11);

3.4 Сложные запросы

– Составление общей таблицы для действующих и бывших спортсменов
with a AS (SELECT project.athletes.first_name, project.athletes.last_name, project.athletes.middle_name, project.athletes.sport_id, project.trainers.first_name, project.trainers.last_name, project.trainers.middle_name
FROM project.athletes LEFT JOIN project.trainers ON project.athletes.sport_id = project.trainers.sport_id

UNION
SELECT project.former_athletes.first_name, project.former_athletes.last_-
name, project.former_athletes.middle_name, project.former_athletes.sport_-
id, project.trainers.first_name, project.trainers.last_name, project.trainers.middle_-
name
FROM project.former_athletes LEFT JOIN project.trainers ON project.former_-
athletes.sport_id = project.trainers.sport_id)

– GROUP BY + HAVING
– Сортировка по дате и городу соревнований по плаванию
SELECT date, city, sport_id FROM project.competitions GROUP BY date,
city, sport_id HAVING sport_id=4;

– ORDER BY
– Сортировка по количеству учеников у тренера (и нынешних и бывших)
SELECT project.trainers.sport_id, COUNT(*) as count
FROM a LEFT JOIN project.trainers ON a.sport_id = project.trainers.sport_-
id ORDER BY count, project.trainers.sport_id;

– <func>(...) OVER(...):
– PARTITION BY
– Сортировка городов, в которых проходили соревнования по количеству
соревнований
SELECT DISTINCT city, COUNT(sport_id) OVER(PARTITION BY city)
AS count FROM project.competitions;

– <func>(...) OVER(...):
– ORDER BY
– Упорядочение расписание по времени начала тренировки
SELECT time, day_of_week, sport_id, ROW_NUMBER() OVER(ORDER
BY time ASC) AS Row_N FROM (
SELECT DISTINCT day_of_week, time, sport_id FROM project.timetable)
AS times
ORDER BY Row_N ASC;

– <func>(...) OVER(...):
PARTITION BY + ORDER BY
– У кого из тренеров больше всего "успешных"спортсменов (выступив-
ших на соревнованиях и получивших какой-то результат)

with sportmen AS (SELECT DISTINCT athlets_id, trainer_id, sport_-
id FROM project.former_athletes WHERE best_competition_id IS NOT
NULL
UNION
SELECT DISTINCT project.athletes.athlets_id, project.athletes.trainer_-
id, project.athletes.sport_id FROM project.athletes LEFT JOIN project.athlete's_-
achievement
ON project.athletes.athlets_id = project.athlete's_achievement.athlets_id
WHERE project.athlete's_achievement.result IS NOT NULL)
SELECT DISTINCT sportmen.trainer_id, sport_id, COUNT(sportmen.athlets_-
id) OVER(PARTITION BY trainer_id) AS count FROM sportmen
ORDER BY count, sportmen.trainer_id;

– <func>(...) OVER(...):
– <func> - все 3 типа функций - агрегирующие, ранжирующие, смеще-
ния
– Для каждого соревнования найти предыдущее по этому виду спорта,
– для каждого соревнования посчитать количество "наших"спорсменов,
которые в них учавствовали
with sporsmen AS (SELECT project.athletes.first_name, project.athletes.last_-
name, project.athletes.middle_name, project.athletes.sport_id,
project.trainers.first_name, project.trainers.last_name, project.trainers.middle_-
name
FROM project.athletes LEFT JOIN project.trainers ON project.athletes.sport_-
id = project.trainers.sport_id
UNION
SELECT project.former_athletes.first_name, project.former_athletes.last_-
name, project.former_athletes.middle_name, project.former_athletes.sport_-
id,
project.trainers.first_name, project.trainers.last_name, project.trainers.middle_-
name
FROM project.former_athletes LEFT JOIN project.trainers ON project.former_-
athletes.sport_id = project.trainers.sport_id),
count_sportmen AS (SELECT athlets_id, result, competition_id FROM
project.athlete's_achievement WHERE result IS NOT NULL
UNION
SELECT athlets_id, best_result, best_competition_id FROM project.former_-
athletes WHERE best_result IS NOT NULL),

competition AS (SELECT *, DENSE_RANK() OVER (ORDER BY sport_-
id) AS rang

```
FROM project.competitions
ORDER BY rang ASC),
counts AS (SELECT DISTINCT competition_id, COUNT(athlets_id) OVER
(PARTITION BY competition_id) AS count_sportman FROM count_-
sportmen)

SELECT competition.competition_id, date, city, competition_lavel, sport_-
id, count_sportman,
LAG(competition.competition_id) OVER (PARTITION BY rang ORDER
BY date) AS last_competition
FROM competition LEFT JOIN counts ON competition.competition_id =
counts.competition_id
ORDER BY date ASC;
```