Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО
Факультет программной инженерии и компьютерной техники

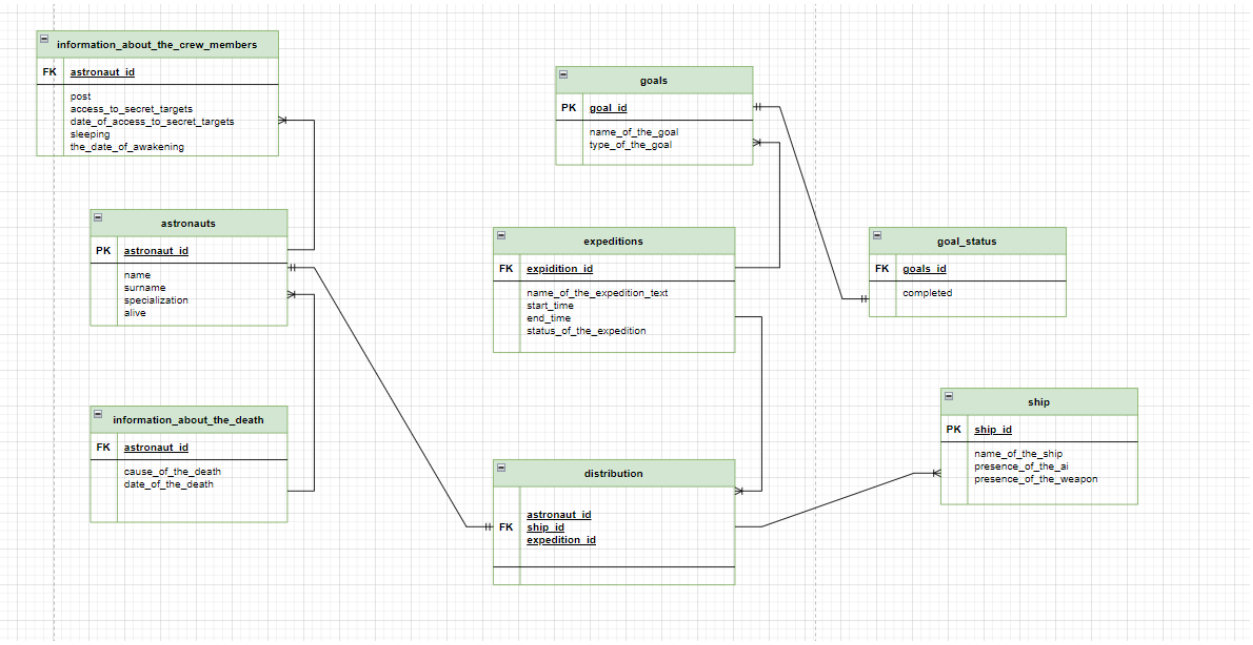
Базы Данных
Работа с Базами Данных
Лабораторная работа №3
Вариант 914

Выполнил:
Студент группы P3116
Брагин Роман Андреевич
Проверил:
Гаврилов Антон Валерьевич

г. Санкт-Петербург

2024 г.

1.Исходная Даталогическая модель:



2.Исходное задание:

Для отношений, полученных при построении предметной области из лабораторной работы №1, выполните следующие действия:

- Опишите функциональные зависимости для отношений полученной схемы (минимальное множество);
- Приведите отношения в 3NF (как минимум). Постройте схему на основеNF (как минимум).
- Опишите изменения в функциональных зависимостях, произошедшие после преобразования в 3NF (как минимум). Постройте схему на основеNF;
- Преобразуйте отношения в BCNF. Докажите, что полученные отношения представлены в BCNF. Если ваша схема находится уже в BCNF, докажите это;
- Какие денормализации будут полезны для вашей схемы? Приведите подробное описание.

Придумайте триггер и связанную с ним функцию, относящиеся к вашей предметной области, согласуйте их с преподавателем и реализуйте на языке PL/pgSQL

3.Функциональные зависимости

ASTRONAUTS: ASTRONAUT_ID → NAME, SURNAME, SPECIALIZATION, ALIVE

INFORMATION_ABOUT_THE_DEATH: ASTRONAUT_ID → CAUSE_OF_THE_DEATH, DATE_OF_THE_DEATH

INFORMATION_ABOUT_THE_CREW_MEMBERS: ASTRONAUT_ID → POST, ACCESS_T0_SECRET_TARGETS, DATE_OF_ACCESS_T0_SECRET_TARGETS, SLEEPING, THE_DATE_OF_AWAKENING

SHIP: SHIP_ID → NAME_OF_THE_SHIP, PRESENCE_OF_THE_AI, PRESENCE_OF_THE_WEAPON

GOALS: GOAL_ID → NAME_OF_THE_GOAL, TYPE_OF_THE_GOAL

GOAL_STATUS: GOALS_ID → COMPLETED

EXPEDITIONS: EXPEDITION_ID → NAME_OF_THE_EXPEDITION_TEXT, START_TIME, END_TIME, STATUS_OF_THE_EXPEDITION, GOAL_ID

DISTRIBUTION: ASTRONAUT_ID, SHIP_ID, EXPEDITION_ID → ()

4.Нормальные формы:

Первая нормальная форма (1NF) — это базовый уровень нормализации базы данных, который в основном касается структуры хранения данных в таблицах. Для того чтобы таблица базы данных соответствовала первой нормальной форме, она должна удовлетворять следующим условиям:

Атомарность значений: Каждое поле таблицы должно содержать атомарное значение, то есть значение, которое не может быть дальше разделено на более мелкие части с точки зрения базы данных. Например, поле, содержащее список телефонных номеров, нарушает правило атомарности, так как список можно разделить на отдельные номера.

Однозначность столбцов: Каждый столбец в таблице должен быть уникальным по названию и содержать данные только одного определённого типа. Например, столбец с названием "Телефон" должен содержать только телефонные номера, а не смешивать номера телефонов с адресами электронной почты.

Однозначное ключевое поле: Каждая строка в таблице должна быть уникально идентифицируема с помощью "ключа" (первичного ключа), который не содержит повторяющихся значений. Это обеспечивает возможность надёжного обращения к каждой записи в таблице.

Порядок данных не имеет значения: Порядок строк и столбцов не должен влиять на способ интерпретации данных. Это означает, что данные в таблице должны оставаться согласованными независимо от того, как строки или столбцы будут переставлены.

Итог: Все таблицы соответствуют первой нормальной форме (1NF). Они содержат атомарные значения, каждая запись уникально идентифицируется, и порядок строк или столбцов не влияет на данные.

Вторая нормальная форма (2NF) — это следующий уровень нормализации в базах данных после первой нормальной формы (1NF). Основное требование 2NF заключается в устранении проблем, связанных с частичной функциональной зависимостью не ключевых атрибутов от первичного ключа. Все не ключевые атрибуты должны полностью зависеть от всего первичного ключа, а не только от его части.

Вывод: Все таблицы соответствуют требованиям второй нормальной формы (2NF), так как в каждой таблице все не ключевые атрибуты полностью зависят от всего первичного ключа и нет частичных зависимостей.

Третья нормальная форма (3NF) — это дополнительный уровень нормализации в базах данных, который следует за второй нормальной формой (2NF). Устранение транзитивных зависимостей: не ключевые атрибуты не должны зависеть от других не ключевых атрибутов, а только от первичных ключей. Транзитивная зависимость в базах данных происходит, когда один не ключевой атрибут зависит от другого не ключевого атрибута, который, в свою очередь, зависит от первичного ключа. Это может привести к избыточности данных и сложностям в обслуживании данных. Для приведения таблицы в 3NF необходимо разбить таблицу так, чтобы каждый не ключевой атрибут был зависим только от первичных ключей.

Вывод: Все таблицы соответствуют третьей нормальной форме (3NF). В них отсутствуют транзитивные зависимости, и все не ключевые атрибуты зависят только от первичных ключей.

5.BCNF:

BCNF (Boyce-Codd Normal Form) — это усиленная версия третьей нормальной формы (3NF). Чтобы таблица соответствовала BCNF, она должна удовлетворять следующим критериям:

Быть в 3NF: Таблица должна уже соответствовать третьей нормальной форме.

Каждая зависимость определяется суперключом: для любой не ключевой функциональной зависимости, первичный ключ должен быть суперключом, т.е. в каждой нетривиальной функциональной зависимости вида $X \rightarrow Y$, X должен быть суперключом. Суперключ — это набор одного или нескольких атрибутов, который может уникально идентифицировать строку в таблице. Первичный ключ является специальным случаем суперключа, который минимален, то есть не может содержать лишних атрибутов и все же сохранять свойство уникальности.

Вывод: Таблицы, согласно определению, удовлетворяют BCNF.

6.Предложения по денормализации

Денормализация может быть полезна для увеличения производительности запросов, особенно при частых операциях чтения:

Объединение ASTRONAUTS и INFORMATION_ABOUT_THE_DEATH:

Уменьшение количества JOIN-операций для запросов, требующих данных об астронавтах и их статусе жизни.

Включение статуса выполнения задачи (COMPLETED из GOAL_STATUS) в таблицу GOALS:

Упрощение запросов, связанных с целями и их статусами, без необходимости присоединения таблицы GOAL_STATUS.

Но эти предложения приведут к нарушению целостности, что не перекроет полученные плюсы, в виде уменьшения JOIN’ов.

7.Триггер:

```
CREATE OR REPLACE FUNCTION update_astronaut_alive_status()
RETURNS TRIGGER AS
$$
BEGIN
    -- Проверяем, было ли изменено поле CAUSE_OF_THE_DEATH или
DATE_OF_THE_DEATH с NULL на не NULL
    IF (OLD.CAUSE_OF_THE_DEATH IS NULL AND NEW.CAUSE_OF_THE_DEATH IS NOT NULL)
OR
       (OLD.DATE_OF_THE_DEATH IS NULL AND NEW.DATE_OF_THE_DEATH IS NOT NULL)
THEN
        -- Обновляем поле ALIVE в таблице ASTRONAUTS на 'Нет' для
соответствующего ASTRONAUT_ID
        UPDATE ASTRONAUTS
        SET ALIVE = 'Нет'
        WHERE ASTRONAUT_ID = NEW.ASTRONAUT_ID;
    END IF;
    RETURN NEW;
END;
```

```
$$
LANGUAGE plpgsql;


CREATE TRIGGER trigger_update_astronaut_alive
AFTER UPDATE ON INFORMATION_ABOUT_THE_DEATH
FOR EACH ROW
EXECUTE FUNCTION update_astronaut_alive_status();
```

Что делает?
Если в данных о смерти изменяется информация с null на новую, в таблице астронавты, мы делаем атрибут alive – нет.

Вывод: Я узнал нормальные формы и научился писать триггеры и функции