ОТЧЕТ
О ВЫПОЛНЕНИИ ИТОГОВОЙ РАБОТЫ ПО КУРСУ
ЯЗЫК SQL

Выполнила:
Студентка 3 курса ОП “Прикладная математика и информатика”
Шашлова Елизавета

Группа: БПМИ224

Описание предметной области

Предметная область базы данных посвящена небесным телам, их характеристикам и взаимосвязям. Она охватывает различные астрономические объекты, такие как звезды, созвездия, планеты, спутники, звездные скопления, туманности и кометы. Также в базе данных учитываются ученые-астрономы, которые открыли эти объекты, их научные руководители и отношения между учеными.

База данных моделирует иерархические связи между этими объектами, например:

- **Звезды** содержат планетарные системы, состоящие из планет.
- **Планеты** имеют спутники.
- **Созвездия** включают звезды, звездные скопления и туманности.
- **Астрономы** связаны древовидной структурой "ученик-научный руководитель".

Кроме этого, в базе данных хранится информация о физических и астрономических характеристиках объектов, таких как спектральный класс звезды, эксцентриситет планет, видимость туманностей, периоды обращения комет и другие данные.

Создание данной базы данных может быть полезно компании по ряду причин:

- Использование реляционной структуры с ограничениями целостности данных исключает вероятность ошибок, таких как дублирование информации, несогласованность данных и нарушение иерархий.

- База данных позволяет прослеживать взаимосвязи между объектами (например, спутники → планеты → звезды → созвездия).

- Астрономы и исследователи могут быстро получать данные, необходимые для анализа небесных объектов и их характеристик.

- База данных может быть расширена, чтобы включать новые небесные тела, новые параметры и дополнительные связи.

База данных содержит 8 атрибутов: небесные тела и ученые, открывшие их, а также 7 связей между сущностями. Все таблицы содержат первичный ключ, у многих он составной, так как многие сущности являются слабыми. Например, планета определяется не одним своим названием, но и уникальным названием звезды, на орбите которой она находится.

Технические требования для базы данных:

- Ввод, модификация и удаление данных о небесных телах, включая: Звезды, Планеты и экзопланеты, Спутники, Туманности, Звездные скопления, Созвездия, Кометы.
- Выполнение поиска данных о людях, связанных с небесными телами (астрономах)
- Формирование отчетов по небесным телам:
    - Спутники и их планеты
    - Планеты и звезды, вокруг которых они вращаются
    - Созвездия и их составные части (звезды, туманности)

Спецификации требований пользователей для представлений таблиц базы данных:

1. Звездные скопления

Требования:

- Ввод данных:
    - Пользователи должны иметь возможность вводить информацию о новых звездных скоплениях.
- Модификация данных:
    - Возможность редактировать существующие данные, такие как видимая звездная величина или вид скопления.
- Удаление данных:
    - Возможность удалить запись о звездном скоплении.
- Поиск и фильтрация:
    - Поиск звездных скоплений по названию или типу.
    - Фильтрация по созвездиям.

Вывод данных:

- Показать все звездные скопления в заданном созвездии.
- Отчет о звездных скоплениях с самой высокой видимой величиной.

2. Звезды

Требования:

- Ввод данных:
    - Пользователи могут добавлять новые звезды с информацией о созвездии, спектральном классе, абсолютной звездной величине, звездном скоплении, первооткрывателе.
- Модификация данных:
    - Обновление данных, таких как спектральный класс или абсолютная звездная величина.

- Удаление данных:
    - Удаление звезд, которые были удалены из каталога.
- Поиск и фильтрация:
    - Поиск звезд по названию.
    - Фильтрация звезд по спектральному классу, созвездию или звездному скоплению.

Вывод данных:

- Показать все звезды в заданном созвездии.
- Отчет о первооткрывателях звезд.

3. Кометы

Требования:

- Ввод данных:
    - Пользователи вводят данные о кометах: название, материнская звезда, период обращения, год открытия.
- Модификация данных:
    - Возможность изменять данные, например, период обращения или год открытия.
- Удаление данных:
    - Удаление комет, данные о которых устарели или некорректны.
- Поиск и фильтрация:
    - Поиск комет по названию или году открытия.
    - Фильтрация по материнской звезде.

Вывод данных:

- Отчет о всех кометах, обращающихся вокруг определенной звезды.
- Отчет по годам открытия комет.

4. Созвездия

Требования:

- Ввод данных:
    - Добавление нового созвездия с его названием, условиями видимости и количеством звезд.
- Модификация данных:
    - Изменение условий видимости или количества звезд.
- Удаление данных:
    - Удаление записи о созвездии, если оно больше не наблюдается.
- Поиск и фильтрация:
    - Поиск созвездий по названию.
    - Фильтрация по условиям видимости.

Вывод данных:

- Отчет о созвездиях с наибольшим количеством звезд.
- Список созвездий, доступных для наблюдения из определенной точки.

5. Туманности

Требования:

- Ввод данных:
    - Ввод информации о туманностях: название, созвездие, год открытия, первооткрыватель.
- Модификация данных:
    - Изменение данных о первооткрывателе или году открытия.
- Удаление данных:
    - Удаление устаревших данных.
- Поиск и фильтрация:
    - Поиск туманностей по названию.
    - Фильтрация по созвездию или первооткрывателю.

Вывод данных:

- Список всех туманностей, обнаруженных в заданном созвездии.
- Отчет о годах открытия туманностей.

6. Планеты и экзопланеты

Требования:

- Ввод данных:
    - Добавление данных о планетах: название, материнская звезда, класс, эксцентриситет, наличие колец, первооткрыватель.
- Модификация данных:
    - Изменение данных о характеристиках планет, например, класса или эксцентриситета.
- Удаление данных:
    - Удаление записей о планетах, существование которых не подтверждено.
- Поиск и фильтрация:
    - Поиск планет по названию.
    - Фильтрация по классу, наличию колец или материнской звезде.

Вывод данных:

- Список всех экзопланет, обнаруженных вокруг заданной звезды.
- Отчет о планетах с эксцентриситетом выше заданного значения.

7. Спутники

Требования:

- Ввод данных:
    - Добавление спутников с указанием их названия, планеты, звезды, эксцентриситета и размера.
- Модификация данных:
    - Изменение характеристик спутника, таких как эксцентриситет или размер.
- Удаление данных:
    - Удаление данных о спутниках, существование которых не подтверждено.
- Поиск и фильтрация:
    - Поиск спутников по названию.
    - Фильтрация по планете или материнской звезде.

Вывод данных:

- Список всех спутников заданной планеты.
- Отчет о спутниках с эксцентриситетом выше заданного значения.

8. Астрономы

Требования:

- Ввод данных:
    - Ввод информации об астрономах: имя, научный руководитель, страна проживания.
- Модификация данных:
    - Изменение данных о научном руководителе или стране проживания.
- Удаление данных:
    - Удаление записи об астрономе.
- Поиск и фильтрация:
    - Поиск астрономов по имени.
    - Фильтрация по стране проживания или руководителю.

Вывод данных:

- Список астрономов из заданной страны.
- Отчет о научных руководителях и их учениках.

Логическое проектирование:

Таблица Star_Clusters (Звездные скопления)

Ключ	**Атрибуты**	**Тип данных**
Primary Key	cluster_name	VARCHAR(100)
	cluster_type	VARCHAR(50)
Foreign Key -> Constellations	constellation_name	VARCHAR(100)
	apparent_magnitude	REAL

Таблица Stars (Звезды)		
Ключ	**Атрибуты**	**Тип данных**
Primary Key	star_name	VARCHAR(100)
	spectral_class	VARCHAR(20)
Foreign Key -> Constellations	constellation_name	VARCHAR(100)
	absolute_magnitude	REAL
Foreign Key -> Star_Clusters	cluster_name	VARCHAR(100)
Foreign Key -> Astronomers	discoverer_name	VARCHAR(100)

Таблица Comets (Кометы)		
Ключ	**Атрибуты**	**Тип данных**
Primary Key	comet_name	VARCHAR(100)
Foreign Key -> Stars	parent_star_name	VARCHAR(100)

	orbital_period	INTEGER
	discovery_year	INTEGER

Таблица Constellations (Созвездия)		
Ключ	**Атрибуты**	**Тип данных**
Primary Key	constellation_name	VARCHAR(100)
	visibility_conditions	VARCHAR(200)
	star_count	INTEGER

Таблица Nebulae (Туманности)		
Ключ	**Атрибуты**	**Тип данных**
	nebula_name	VARCHAR(100)
Foreign Key -> Constellations	constellation_name	VARCHAR(100)
	discovery_year	INTEGER
Foreign Key -> Astronomers	discoverer_name	VARCHAR(100)

Таблица Planets_Exoplanets (Планеты и экзопланеты)		
Ключ	**Атрибуты**	**Тип данных**
Primary Key	planet_name	VARCHAR(100)
Foreign Key -> Stars	parent_star_name	VARCHAR(100)
	class	VARCHAR(100)
	eccentricity	REAL

	has_rings	BOOLEAN
Foreign Key -> Astronomers	discoverer_name	VARCHAR(100)

Таблица Satellites (Спутники)		
Ключ	**Атрибуты**	**Тип данных**
Primary Key	satellite_name	VARCHAR(100)
Foreign Key -> Planets_Exoplanets	planet_name	VARCHAR(100)
Foreign Key -> Stars	star_name	VARCHAR(100)
	eccentricity	REAL
	size_relative_to_planet_radius	REAL

Таблица Astronomers (Астрономы)		
Ключ	**Атрибуты**	**Тип данных**
Primary Key	scientist_name	VARCHAR(100)
Foreign Key -> Astronomers	supervisor_name	VARCHAR(100)
	country_of_residence	VARCHAR(100)

Типы связей между таблицами

1. **Связь "Созвездия - Звезды"**:

   1:M — Одно созвездие может содержать много звезд.

2. **Связь "Звезды - Планеты"**:

1:M — Одна звезда может иметь множество планет.

3. **Связь "Планеты - Спутники"**:

   1:M — У одной планеты может быть много спутников.

4. **Связь "Звезды - Кометы"**:

   1:M — Одна звезда может быть родительской для множества комет.

5. **Связь "Созвездия - Звездные скопления"**:

   1:M — Одно созвездие может включать множество звездных скоплений.

6. **Связь "Созвездия - Туманности"**:

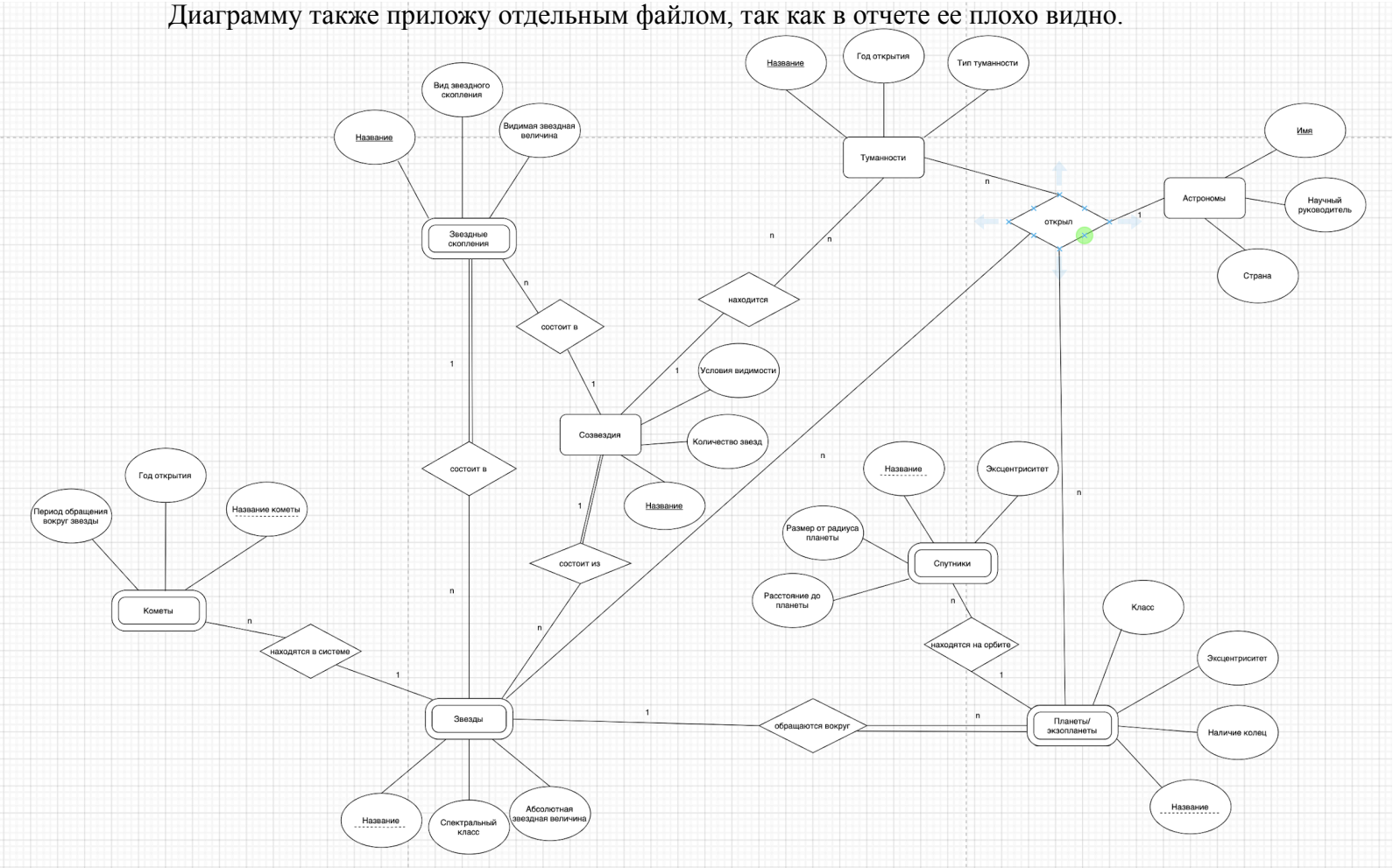
   1:M — Одно созвездие может содержать множество туманностей.

7. **Связь "Астрономы - Открытия"**:

   1:M — Один астроном может быть открывателем множества объектов (звезды, планеты, туманности и т.д.).

ER-диаграмма базы данных:

Диаграмму также приложу отдельным файлом, так как в отчете ее плохо видно.



Физическое проектирование и типичные запросы:

```sql
-- Создадим таблицы --
CREATE TABLE constellations (
    constellation_name VARCHAR(100) PRIMARY KEY,
    visibility_conditions VARCHAR(100),
    number_of_stars INT
);

-- В таблице астрономов будет наблюдаться древовидная структура, заданная списком смежности --
CREATE TABLE astronomers (
    scientist_name VARCHAR(100) PRIMARY KEY,
    country_of_residence VARCHAR(100),
    supervisor_name VARCHAR(100) REFERENCES astronomers(scientist_name)
);

CREATE TABLE star_clusters (
    cluster_name VARCHAR(100) PRIMARY KEY,
    apparent_magnitude NUMERIC(10, 3),
    constellation_name VARCHAR(100) REFERENCES constellations(constellation_name),
    cluster_type VARCHAR(50) CHECK(cluster_type IN ('Шаровое', 'Рассеянное'))
);

CREATE TABLE stars (
    star_name VARCHAR(100) PRIMARY KEY,
    constellation_name VARCHAR(100) REFERENCES constellations(constellation_name),
    spectral_class VARCHAR(20),
    absolute_magnitude NUMERIC(10, 3),
    cluster_name VARCHAR(100) REFERENCES star_clusters(cluster_name),
    discoverer_name VARCHAR(100) REFERENCES astronomers(scientist_name)
);
```

```sql
CREATE TABLE comets (
    comet_name VARCHAR(100),
    parent_star_name VARCHAR(100),
    orbital_period INT,
    discovery_year INT,
    PRIMARY KEY (comet_name, parent_star_name),
    FOREIGN KEY (parent_star_name) REFERENCES stars(star_name)
);

CREATE TABLE nebulae (
    nebula_name VARCHAR(100),
    constellation_name VARCHAR(100),
    discovery_year INT,
    discoverer_name VARCHAR(100) DEFAULT NULL REFERENCES astronomers(scientist_name),
    PRIMARY KEY (nebula_name, constellation_name),
    FOREIGN KEY (constellation_name) REFERENCES constellations(constellation_name)
);
```

```sql
CREATE TABLE planets (
    planet_name VARCHAR(100),
    parent_star_name VARCHAR(100),
    class VARCHAR(100),
    eccentricity NUMERIC(10, 3),
    has_rings BOOLEAN CHECK(has_rings IN (true, false)),
    discoverer_name VARCHAR(100) DEFAULT 'Неизвестно',
    PRIMARY KEY (planet_name, parent_star_name),
    FOREIGN KEY (parent_star_name) REFERENCES stars(star_name),
    FOREIGN KEY (discoverer_name) REFERENCES astronomers(scientist_name)
);

CREATE TABLE satellites (
    satellite_name VARCHAR(100),
    planet_name VARCHAR(100),
    star_name VARCHAR(100),
    eccentricity NUMERIC(10, 3),
    size_relative_to_planet NUMERIC(10, 3),
    PRIMARY KEY (satellite_name, planet_name, star_name),
    FOREIGN KEY (planet_name, star_name) REFERENCES planets(planet_name, parent_star_name)
);
```

Примеры запросов к БД:

1. Найти звезды в созвездиях, видимых круглый год.

```sql
SELECT star_name
FROM stars
JOIN constellations
ON stars.constellation_name = constellations.constellation_name
WHERE constellations.visibility_conditions = 'All year round';
```

2. Найти звездные скопления с максимальной видимой звездной величиной.

```sql
SELECT cluster_name
FROM star_clusters
WHERE apparent_magnitude = (
    SELECT MAX(apparent_magnitude)
    FROM star_clusters
);
```

3. Найти туманности, открытые учеными из Франции.

```sql
SELECT nebula_name
FROM nebulae
WHERE discoverer_name IN (
    SELECT scientist_name
    FROM astronomers
    WHERE country_of_residence = 'France'
);
```

4. Вывести все объекты, входящие в Солнечную систему.

```sql
SELECT comet_name AS object_name FROM comets
WHERE parent_star_name = 'Sun'
UNION
SELECT planet_name AS object_name FROM planets
WHERE parent_star_name = 'Sun'
UNION
SELECT satellite_name AS object_name FROM satellites
WHERE star_name = 'Sun';
```

5. Найти ученого с максимальным количеством открытий.

```sql
SELECT scientist_name
FROM astronomers
JOIN (
    SELECT DISTINCT planet_name AS body_name, discoverer_name FROM planets
    UNION
    SELECT DISTINCT nebula_name AS body_name, discoverer_name FROM nebulae
    UNION
    SELECT DISTINCT star_name AS body_name, discoverer_name FROM stars
) AS discoveries
ON astronomers.scientist_name = discoveries.discoverer_name
GROUP BY astronomers.scientist_name
ORDER BY COUNT(*) DESC
LIMIT 1;
```

6. Вывести список планет и их спутников.

```sql
SELECT planets.planet_name, satellites.satellite_name
FROM planets
JOIN satellites
ON planets.planet_name = satellites.planet_name;
```

7. Найти ученых, открывших хотя бы одну планету или туманность.

```sql
SELECT DISTINCT scientist_name
FROM astronomers
WHERE scientist_name IN (
    SELECT discoverer_name FROM nebulae
) OR scientist_name IN (
    SELECT discoverer_name FROM planets
);
```

8. Вывести названия планет, входящих в планетарные системы звезд, которые находятся в созвездиях с шаровыми звездными скоплениями.

```sql
SELECT planet_name, parent_star_name
FROM planets
WHERE parent_star_name IN (
    SELECT star_name
    FROM stars
    WHERE constellation_name IN (
        SELECT constellation_name
        FROM star_clusters
        WHERE cluster_type = 'Globular'
    )
);
```

9. Найти звезды в осенних созвездиях с планетарными системами, содержащими планеты с эксцентриситетом больше 0.1.

```sql
SELECT star_name
FROM stars
WHERE constellation_name IN (
    SELECT constellation_name
    FROM constellations
    WHERE visibility_conditions = 'Autumn'
) AND star_name IN (
    SELECT parent_star_name
    FROM planets
    WHERE eccentricity > 0.1
);
```

10. Добавить столбец с расстоянием до планеты в таблицу спутников.

```sql
ALTER TABLE satellites ADD COLUMN distance_to_planet INT;
```

11. Заполнить данные в столбце расстояния до планеты.

```sql
UPDATE satellites
SET distance_to_planet = 384000
WHERE satellite_name = 'Moon' AND star_name = 'Sun' AND planet_name = 'Earth';

UPDATE satellites
SET distance_to_planet = 436000
WHERE satellite_name = 'Titania' AND star_name = 'Sun' AND planet_name = 'Uranus';

UPDATE satellites
SET distance_to_planet = 330000
WHERE satellite_name = 'Triton' AND star_name = 'Sun' AND planet_name = 'Neptune';

UPDATE satellites
SET distance_to_planet = 55000000
WHERE satellite_name = 'Nereid' AND star_name = 'Sun' AND planet_name = 'Neptune';

UPDATE satellites
SET distance_to_planet = 12700
WHERE satellite_name = 'Hippocamp' AND star_name = 'Sun' AND planet_name = 'Neptune';
```

12. Вычислить средний размер спутников относительно планет.

```sql
SELECT planet_name, AVG(size_relative_to_planet) AS avg_satellite_size
FROM satellites
GROUP BY planet_name;
```

13. Посчитать максимальное и минимальное достижимое расстояние между спутниками Нептуна.

```sql
SELECT S1.satellite_name, S2.satellite_name,
    (S1.distance_to_planet + S2.distance_to_planet) AS max_distance,
    ABS(S1.distance_to_planet - S2.distance_to_planet) AS min_distance
FROM satellites AS S1
CROSS JOIN satellites AS S2
WHERE S1.satellite_name != S2.satellite_name
AND S1.planet_name = 'Neptune'
AND S2.planet_name = 'Neptune'
ORDER BY S1.satellite_name;
```