СУБД: Fauna
Модель СУБД: Multi-model

a. История развития СУБД
Fauna была создана бывшими сотрудниками Twitter для решения проблем, связанных с масштабированием традиционных баз данных. Была создана в качестве стартапа выходцами из команды разработчиков этой социальной сети. Она использует ресурсы облачных провайдеров и реализует модель бессерверных вычислений – то есть, оплаты не за арендованный сервер или дисковое пространство, а только за конкретно использующиеся в каждый момент времени ресурсы. Система представлена в 2017 год.

b. Инструменты для взаимодействия с СУБД
Fauna предоставляет RESTful API, GraphQL API, а также клиентские библиотеки на различных языках программирования, таких как JavaScript, Java, C#, Python, Go и так далее.

c. Database engine
Fauna использует собственный разработанный engine, оптимизированный для облачных вычислений и мульти-модельных данных, поддерживающий транзакции с сериализуемой изоляцией.

d. Язык запросов
Fauna использует язык запросов FQL (Fauna Query Language), который предоставляет гибкость при работе с данными и поддержку транзакционных операций.
Основные характеристики FQL:

1) Выразительность: FQL предоставляет набор операций для чтения, создания, обновления и удаления данных, а также поддерживает сложные запросы, включая условия, циклы и рекурсию.
2) Функциональный стиль: FQL имеет функциональную природу, где запросы строятся как выражения, использующие друг друга в качестве аргументов.
3) Транзакционность: Каждый запрос в FQL может быть выполнен в контексте транзакции, обеспечивая ACID-свойства даже в распределенной среде.
4) Интеграция с разными моделями данных: благодаря мульти-модельности Fauna, FQL позволяет эффективно работать как с документами, так и с графовыми структурами данных.

Например:
1) Создание базы данных:

```
Create(Collection("Users"), {
  data: {
    name: "Darth Vader",
    email: "Darth.Vader@example.com",
    active: true
  }
})
```

Этот запрос создает новый документ в коллекции "Users" с заданными атрибутами.

2) Чтение данных

```
Get(Match(Index("users_by_email"), "Darth.Vader@example.com"))
```

Здесь запрос ищет пользователя по индексу "users_by_email", который возвращает документ для указанного адреса электронной почты.

3) Удаление данных

```
Delete(Ref(Collection("Users"), "user_id"))
```

Удаляет документ пользователя по его ID.

4) Обновление данных

```
Update(Ref(Collection("Users"), "user_id"), {
  data: { active: false }
})
```

Этот запрос обновляет атрибут "active" у пользователя с указанным ID.

e. Распределение файлов БД по разным носителям
Fauna как облачная СУБД автоматически управляет распределением данных на физических и виртуальных носителях, обеспечивая оптимальную производительность и надежность.

f. Языки программирования
Система написана на смеси языков, включая JavaScript/TypeScript, Python, Java, Go, Ruby, Scala

g. Типы индексов
Fauna поддерживает создание пользовательских индексов для улучшения производительности запросов. Индексы могут быть созданы по любому набору полей в документах.
Пример:

- Основные индексы: создаются для быстрого доступа к документам по ключу или уникальным атрибутам.
- Составные индексы: могут включать несколько полей документа, что позволяет эффективно выполнять сложные запросы, включающие множественные условия.
- Индексы сортировки: поддерживают сортировку данных по одному или нескольким полям.
- Геоиндексы: предназначены для работы с географическими данными и запросами, основанными на местоположении.

Пример создания индекса:

```
CreateIndex({
  name: "users_by_email",
  source: Collection("Users"),
  terms: [{ field: ["data", "email"] }],
  unique: true
})
```

Этот индекс позволяет быстро находить пользователя по адресу электронной почты.

h. Процесс выполнения запросов
Запросы обрабатываются с использованием распределенного движка, который оптимизирует выполнение для обеспечения быстродействия и согласованности данных.

i. План запросов
Fauna предоставляет инструменты для анализа выполнения запросов, позволяя разработчикам видеть, как запросы интерпретируются и оптимизируются движком базы данных.

j. Транзакции
Fauna поддерживает полноценные ACID-транзакции даже в распределенной среде, что является одной из её ключевых особенностей.

k. Методы восстановления
Fauna предоставляет возможности точечного восстановления и резервного копирования данных, что позволяет обеспечить высокую надежность данных.

l. Шардинг
Fauna автоматически распределяет данные между серверами и регионами для оптимизации производительности и надежности. Данные автоматически реплицируются и распределяются таким образом, чтобы минимизировать задержку доступа и максимизировать доступность. Это обеспечивает глобальную масштабируемость без необходимости вручную управлять процессом шардинга.

m. Data Mining, Data Warehousing и OLAP
Fauna поддерживает сложные запросы и агрегации, которые могут использоваться для извлечения полезной информации из данных. Однако, в отличие от классических платформ Data Mining, Fauna не включает встроенных инструментов для автоматического анализа данных или машинного обучения. Fauna не является классическим решением для Data Warehousing, поскольку она ориентирована на оперативную обработку транзакций с высокой производительностью и низкой задержкой. Однако благодаря своим возможностям глобального распределения и масштабируемости, Fauna может использоваться как компонент в более крупной архитектуре данных, где она обеспечивает активное хранилище операционных данных, которые затем могут экспортироваться или реплицироваться в специализированные системы для аналитики. Fauna поддерживает множество типов запросов, включая агрегации и сложные многотабличные запросы, которые необходимы для OLAP-подобных анализов.

n. Методы защиты
Fauna предлагает шифрование данных в покое и в транзите, а также поддерживает различные модели авторизации для управления доступом.

o. Сообщества и разработчики
Fauna развивается компанией Fauna Inc., с участием сообщества разработчиков, которые могут вносить предложения через публичные репозитории и форумы.

p, q. Создание и использование демобазы
Для демонстрации работы можно создать примерные данные и использовать их с помощью FQL. Демонстрационная база и обучающие материалы часто доступны на сайте Fauna.

Частично данный пункт был описан выше. Не буду повторяться, но добавлю новую информацию.
Создание коллекции пльзователей

```
CreateCollection({ name: "Users" })
```

Запрос на чтение:

```
Get(Match(Index("users_by_email"), "Simpson@example.com"))
```

Запрос на обновление:

```
Update(Select("ref", Get(Match(Index("users_by_email"), "Simpson@example.com"))), {
  data: { age: 40 }
})
```

Запрос на создание новых юзеров

```
Create(Collection("Users"), {
  data: {
    name: "Saruman",
    age: 2000,
    email: "Saruman@example.com"
  }
})
Create(Collection("Users"), {
  data: {
    name: "Gendalf",
    age: 2500,
    email: "Gendalf@example.com"
  }
})
```

r. Документация и обучение
Документация и учебные материалы доступны на официальном сайте Fauna, включая подробные руководства, примеры кода и видеоуроки. Fauna предоставляет бета-версию.
Документация:
https://docs.fauna.com/fauna/current/

s. Быть в курсе событий
Для отслеживания новостей можно подписаться на рассылку Fauna, следить за обновлениями на сайте и в социальных сетях.