

Общая база данных.

Общая база данных - это стиль интеграции, при котором различные системы и приложения используют одну и ту же базу данных для хранения и обмена данными. Вместо того чтобы каждая система имела свою собственную базу данных и передавала данные между ними через файлы, API или другие механизмы, они все обращаются к одной общей базе. Чаще всего используется в рамках одной компании, то есть между системами в рамках одного какого-то большого IT-сервиса (потому что общая внешняя база данных — это небезопасно).

Как работает интеграция через общую базу данных:

1. Вначале определяются структура и схема общей базы данных, которые будут использоваться для хранения и обмена данными между системами. Схема должна быть разработана таким образом, чтобы удовлетворять требованиям всех интегрируемых систем. (то есть решается вопрос, какие таблицы с какими данными нужны)
2. Различные системы и приложения настраиваются для доступа к общей базе данных. Это может потребовать изменения существующих систем или создания новых интерфейсов для работы с базой данных. (но это проще, чем настроить интерфейс API)
3. Системы выполняют операции чтения и записи данных в общую базу данных, следуя определенным правилам и протоколам, чтобы обеспечить целостность и согласованность данных.
4. В процессе работы системы могут выполнять запросы к общей базе данных для получения или обновления информации, которая им необходима для выполнения своих функций. Данные в базе обновляются в реальном времени, что позволяет системам оперативно реагировать на изменения.

Если рассматривать подробнее, то вот как это может выглядеть:

Шаг 1: Определение требований к базе данных

- 1.1 Собрать команду из представителей всех отделов, которые будут использовать базу.
- 1.2 Составить список всех типов данных, которые будут храниться (например, клиенты, заказы, товары).
- 1.3 Определить, как эти данные будут связаны между собой.

Шаг 2: Проектирование схемы базы данных

- 2.1 Разработать структуру каждой таблицы: поля, их типы и связи между таблицами.
- 2.2 Подготовить диаграмму схемы базы для визуализации.

Шаг 3: Настройка сервера базы данных

- 3.1 Выбрать сервер для базы данных (может быть физический сервер или облачный).
- 3.2 Установить программное обеспечение для управления базой данных (например, MySQL, PostgreSQL, etc.).
- 3.3 Запустить сервер и создать новую базу данных, используя ранее разработанную схему.

Шаг 4: Создание пользователей и настройка прав доступа

- 4.1 В системе управления базой данных создать аккаунты для каждого отдела.
- 4.2 Установить разрешения: какие таблицы каждый аккаунт может просматривать, модифицировать, создавать или удалять.

Шаг 5: Интеграция существующих систем с базой данных

- 5.1 Установить драйверы и библиотеки для работы с базой данных на серверах или компьютерах, где работают системы отделов.
- 5.2 В коде каждой системы написать функции для подключения к базе, а также функции для чтения и записи данных.
- 5.3 Провести первичное заполнение таблиц данными, если это необходимо.

Шаг 6: Тестирование

- 6.1 Симулировать типичные операции для каждого отдела (создание заказа, изменение наличия товара и т.д.).
- 6.2 Проверить, правильно ли эти действия отражаются в базе данных.

Шаг 7: Мониторинг и обслуживание

- 7.1 Установить систему мониторинга для отслеживания загрузки и доступности базы данных.
- 7.2 Настроить регулярное резервное копирование базы данных.

Шаг 8: Обновление и масштабирование

- 8.1 При необходимости добавить новые поля или таблицы в схему базы данных.
- 8.2 При росте нагрузки добавить дополнительные ресурсы (например, дополнительные серверы или увеличение мощности текущего).

Это базовые шаги, которые нужно предпринять для интеграции разных систем через общую базу данных. Каждый шаг может включать в себя множество подшагов и требовать определенных технических навыков.

Примеры использования общей базы данных:

- В компаниях, предоставляющих техническую поддержку клиентов, общая база данных может использоваться для хранения информации о заявках, сотрудниках и клиентах. В этом случае разные системы (например, CRM, биллинг и система управления заявками) могут иметь доступ к общей базе данных для получения актуальной информации и координации работы.
- Банки и финансовые организации могут использовать общую базу данных для управления информацией о счетах, транзакциях и клиентах. В этом случае системы, такие как интернет-банкинг, системы кредитования и внутренние аналитические системы, могут обращаться к одной и той же базе данных для обработки и анализа финансовых данных одного клиента.

Преимущества интеграции через базу данных

1. Простота реализации: Иногда проще настроить одну общую базу, чем интегрировать множество разных API.
2. Централизация данных: Все данные хранятся в одном месте, что упрощает управление и обслуживание.
3. Согласованность данных: Поскольку все системы используют одну и ту же базу, вероятность несогласованности данных снижается.
4. Быстрый доступ к данным: Системы могут мгновенно получить доступ к актуальной информации.
5. Облегчение аналитики: С централизованными данными проще проводить анализ и отчетность.
6. Экономия ресурсов: Избегается дублирование данных и соответствующие затраты на хранение.

7. Упрощение бэкапа: Одна база данных проще резервировать и восстанавливать.
8. Гибкость: Можно легко добавлять новые системы, подключая их к уже существующей базе данных.
9. Уменьшение времени на интеграцию: Отсутствие необходимости в разработке и поддержке API ускоряет процесс интеграции.
10. Снижение ошибок: Меньше шансов на ошибки при передаче данных, поскольку этап передачи отсутствует.
11. Простота масштабирования: Проще масштабировать одну базу данных, чем множество разнородных систем.

Недостатки интеграции через базу данных

1. Точка отказа: Одна база данных — это единая точка отказа. Если база данных "падает", это затрагивает все подключенные системы.
2. Проблемы с безопасностью: Трудно контролировать доступ к данным на уровне отдельных систем.
3. Зависимость от схемы: Все системы становятся зависимыми от одной схемы базы данных, что затрудняет изменения и обновления.
4. Проблемы с производительностью: Высокая нагрузка на одну базу может замедлить все подключенные системы.
5. Сложность управления версиями: Обновления и изменения в структуре базы могут повлиять на все подключенные системы.
6. Ограниченная гибкость: Невозможно настроить индивидуальные опции хранения и обработки данных для разных систем.
7. Трудности с распределенными системами: Если системы физически расположены в разных местах, интеграция через общую базу данных становится сложнее.
8. Сложность миграции: Переход на другую технологию или структуру базы может быть крайне трудоемким.
9. Конфликты транзакций: Сложнее управлять транзакциями и блокировками в многопользовательской среде.
10. Высокие требования к администрированию: Необходим специализированный персонал для обслуживания базы данных.
11. Сложности с аудитом и трассировкой: Отследить, какая система вносила изменения, может быть затруднительно.

В целом, интеграция через общую базу данных может быть эффективным решением в определенных ситуациях, когда системы имеют схожие требования к данным и необходимо обеспечить быстрый доступ к актуальной информации. Однако, в более сложных и динамичных сценариях, где требуются высокая производительность, гибкость и надежность, другие стили интеграции, такие как API или обмен сообщениями через брокеров, могут быть более предпочтительными и эффективными.