Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРТСВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ»

**МИГРАЦИЯ ДАННЫХ И ОБЪЕКТОВ ИЗ ОДНОЙ СУБД В ДРУГУЮ**

Выполнила:

Савицкая Варвара Николаевна

студентка 2 курса 4 группы

факультета ФИТ

Минск, 2024

**Введение**

Миграция проводится по четкому плану, чтобы процесс проходил организованно, непрерывно. Отсутствие планирования часто приводит к непредвиденным простоям, серьезным ошибкам в бизнес-процессах, иногда даже к потерям критических данных. Внешне это выглядит как недоработанная программа с регулярными «глюками» и периодическими отказами. Поэтому план обязателен.

Процесс миграции условно делят на два этапа – подготовку и непосредственно миграцию.

При полн миграция баз данных переносится полностью из источника в приемник.

Версионную миграциючаще выбирают в случаях, когда необходимо контролировать версии программного обеспечения и, если над разработкой программного обеспечения трудится команда специалистов, а не отдельный разработчик. При переносе данных важно соблюдать последовательность, все SQL-запросы должны выполняться один раз. Если не придерживаться этих условий, то можно столкнуться с потерей данных и потратить время на их восстановление.

**Раздел 1. Планирование**

Прежде чем приступать к миграции, необходимо ответить на несколько вопросов:

- Зачем нужна миграция базы данных?

- Какой подход к миграции следует использовать в проекте?

- Собираемся переходить от одной парадигмы к другой?

- Дает ли новая база данных необходимую поддержку того, что используеся сейчас в старой?

Ответы на эти вопросы помогут определить стратегию миграции и выбрать подходящие инструменты.

Причины миграции могут быть различными:

- Уменьшение задержек при вводе-выводе данных.

- Снижение стоимости поддержки.

- Улучшение производительности и масштабируемости.

Рассмотрим конкретный пример. Предположим, есть компания TechSolutions, которая использует MySQL для хранения данных о клиентах, заказах и продуктах. Цель компании — предоставлять высококачественные технологические решения для бизнеса. Количество покупателей более 100.000. Из-за роста объема данных и необходимости улучшения производительности, компания решила мигрировать на PostgreSQL. Причина миграции — улучшение производительности и масштабируемости.

Ответы на вопросы:

1. Зачем нужна миграция базы данных?

Миграция базы данных необходима для улучшения производительности и масштабируемости. В данном случае, компания TechSolutions решила мигрировать с MySQL на PostgreSQL из-за роста объема данных и необходимости улучшения производительности.

2. Какой подход к миграции следует использовать в проекте?

Обычно рекомендуется использовать поэтапный подход, включающий планирование, тестирование, перенос данных и проверку целостности данных.

3. Собираемся переходить от одной парадигмы к другой?

Нет, в данном случае речь идет о миграции с одной реляционной базы данных (MySQL) на другую реляционную базу данных (PostgreSQL). Это не переход от одной парадигмы к другой, а переход между двумя системами управления базами данных в рамках одной парадигмы.

4. Дает ли новая база данных необходимую поддержку того, что используется сейчас в старой?

Да, PostgreSQL предоставляет необходимую поддержку для тех задач, которые выполняются в MySQL.

**Раздел 2. Проверка концепции**

Если обе базы данных используют одну и ту же парадигму (например, реляционные базы данных), миграция будет проще. В этом случае структуры данных и типы данных будут схожими, и различия можно будет легко устранить.

Если базы данных используют разные парадигмы (например, переход с документно-ориентированной базы данных на колоночную), процесс миграции будет сложнее. В этом случае необходимо тщательно продумать моделирование данных и типы запросов.

**Раздел 3. Тестирование**

После проверки концепции необходимо провести тестирование. Рекомендуется начать с тестирования 1% и 5% данных в промежуточной среде. Это поможет выявить возможные проблемы на ранних этапах и убедиться, что моделирование данных и серверная часть работают корректно.

Для компании TechSolutions тестирование будет выглядеть следующим образом:

1. Тестирование 1% данных:

- Экспорт данных: экспорт 1% данных из MySQL в формате CSV. CSV — текстовый формат, предназначенный для табличных данных (листинг 3.1).

SELECT FROM table\_name LIMIT 1000 INTO OUTFILE 'C:/db /file.csv';

Листинг 3.1 – скрипт для экспорта 1% данных из MySQL

- Импорт данных: импорт 1% данных в PostgreSQL с использованием pgloader, pgloader загружает данные в PostgreSQL и позволяет реализовать непрерывную миграцию из текущей базы данных в PostgreSQL (листинг 3.2).

mysql://user:password@host/database\_name postgresql://user:password@host/database\_name --limit 1000

Листинг 3.2 – скрипт для импорта 1% данных в PostgreSQL

**Раздел 4. Стратегии миграции**

Существует две основные стратегии миграции: горячая миграция и холодная миграция.

Холодная миграция предполагает остановку системы на время миграции. Этот подход используется в малых и средних компаниях и обычно выполняется в ночное время, когда система менее востребована.

Контрольный список для холодной миграции:

- Планирование: определить время и ответственных лиц;

- Извлечение: выгрузить все данные;

- Трансформация: преобразовать данные в формат новой базы данных;

- Загрузка: загрузить данные в новую базу данных;

- Проверка: убедиться, что данные совпадают;

- Тестирование: проверить работу системы с новой базой данных;

- Активация: отключить старую базу данных.

Горячая миграция предполагает одновременную запись данных в обе базы данных. Этот подход используется в крупных компаниях, где необходимо минимизировать простои.

Контрольный список для горячей миграции:

- Планирование;

- Репликация: синхронизировать данные между базами данных;

- Тестирование и проверка: проверка данных, которые переносятся в процессе репликации;

- Минимизация простоев;

- Активация: отключение старой базы данных.

Рассмотрим миграцию на примере компании TechSolutions.

1. Подготовка:

- Анализ данных и объектов: идентификация всех таблиц, индексов, представлений и хранимых процедур.

Скрипт выводит список всех таблиц, индексов, представлений и всех хранимых процедур в базе данных MySQL (листинг 4.1).

SHOW TABLES;

SHOW INDEX FROM table\_name;

SHOW FULL TABLES IN database\_name WHERE TABLE\_TYPE LIKE 'VIEW'

SHOW PROCEDURE STATUS WHERE Db = 'database\_name';

Листинг 4.1 – скрипт для экспорта данных из MySQL

2. Выполнение:

- Экспорт данных: экспорт данных из MySQL в формате CSV производится также как это выполнялось в разделе с тестирования (листинг 4.2).

SELECT FROM table\_name INTO OUTFILE 'C:/db /file.csv';

Листинг 4.2 – скрипт для экспорта данных из MySQL

- Преобразование данных: преобразование структуры данных для соответствия требованиям PostgreSQL. Скрипт, изображенный на листинге 4.3 создает новую таблицу в PostgreSQL.

CREATE TABLE table\_name (

id PRIMARY KEY,

column1 VARCHAR(100),

column2 INT,

...

);

Листинг 4.3 – скрипт создания таблице в PostgreSQL.

- Импорт данных: импорт данных в PostgreSQL с использованием pgloader.

3. Пост-миграционные задачи:

- Тестирование: проверка корректности работы всех объектов и данных в PostgreSQL. Скрипт выведет первые 10 строк из указанной таблицы (листинг 4.4)

SELECT FROM table\_name LIMIT 10;

Листинг 4.4 – скрипт вывода 10 строк из таблицы table\_name.

- Оптимизация: создание индексов и настройка параметров PostgreSQL для оптимальной работы. Скрипт создает новый индекс для указанной таблицы и столбца в PostgreSQL (листинг 4.5).

CREATE INDEX index\_name ON table\_name (column\_name);

Листинг 4.5 – скрипт создания индексов

**Заключение**

Миграция данных и объектов из одной СУБД в другую — это сложный, но необходимый процесс для многих организаций. Тщательное планирование, выполнение и пост-миграционные задачи играют ключевую роль в успешной миграции. Использование правильных инструментов и методов позволяет минимизировать риски и обеспечить успешный переход на новую платформу.