Лекция #9

Базы данных

Михаил Моисеев

Внутреннее устройство сервера БД
Оптимизация запросов

Структура сервера БД

- Подсистема взаимодействия с клиентским приложением – управление соединениями с клиентами
- Подсистема синтаксического разбора запросов – компиляция запросов
- Подсистема планирования выполнения запросов – составление планов, анализ статистики
- Подсистема выполнения транзакций – выполнение и откат транзакций, выполнение триггеров и процедур
- Подсистема управления памятью – подкачка необходимых страниц в память, синхронизация данных в памяти и на диске

Взаимодействие с клиентами

Для СУБД Firebird имеются две архитектуры

- CS – для каждого клиента создается отдельный процесс, со своими ресурсами, хорошо распараллеливается на многопроцессорных или многоядерных системах, эффективен при небольшом, заранее известном числе клиентов
- SS – ресурсы разделяются между всеми клиентами, эффективен при большом числе часто подключающихся клиентов, выполняющих «легкие» запросы

Структура файла БД

БД – один или несколько файлов. Файл – набор страниц фиксированного размера (1-16 Кб), все операции осуществляются постранично.

Типы страниц

- страница заголовка БД,
- страница с информацией о распределении страниц,
- страница учета транзакций,
- страница указателей,
- страница данных,
- страница генераторов,
- страница индексов,
- страница для хранения BLOB-данных,
- пустая страница.

Состояние транзакций

- ***Подтвержденное***: транзакция успешно завершила всю свою работу (COMMIT или ROLLBACK при отсутствии изменений)
- ***Отмененное***: транзакция завершилась ROLLBACK, при этом необходимо удалить произведенные ею изменения
- ***Активное***: транзакция не стартовала, стартовала, но еще не завершилась, в том числе из-за отсоединения клиента
- ***Limbo***: транзакция, стартовавшая в режиме 2PC (two phase commit)

Каждая транзакция имеет уникальный идентификатор.

Старейшая заинтересованная транзакция - старейшая транзакция, которая конкурирует с одной из активных транзакций

Старейшая активная транзакция - старейшая транзакция, которая была активной, когда началась старейшая активная в текущий момент транзакция

Версионирование записей

При изменении, добавлении, удалении записи создается ее новая версия, каждая версия имеет идентификатор создавшей ее транзакции

Цель версионирования – обеспечение возможности отката транзакций, а также работы SNAPSHOT-транзакций

Версии записи могут быть подтвержденными и неподтвержденными

Каждая запись имеет по крайней мере одну подтвержденную версию

Транзакция READ COMMITED с параметрами:

- REC VERSION;
- NO REC VERSION.

Сборка мусора

Сборка мусора – процесс удаления версий, которые были созданы отмененными транзакциями и ненужных старых версии записей

Когда изменяющая записи транзакция подтверждается, и все конкурирующие транзакции также завершаются, старая версия становится ненужной

При чтении записи выполняется

- выдача правильной версии записи запросившей ее транзакции
- удаление всех версий, являющиеся мусором

Все транзакции участвуют в сборке мусора.

Индексы

Индекс – объект БД создаваемый специально для повышения производительности запросов

Индекс обеспечивает эффективное упорядочивание записей таблицы по заданному полю (или полям) для указанного порядка

Индексы в Firebird являются однонаправленными

Типы индексов

- кластерный – определят физический порядок записей
- некластерный – указатели на записи в таблице, в соответствии с заданной последовательностью

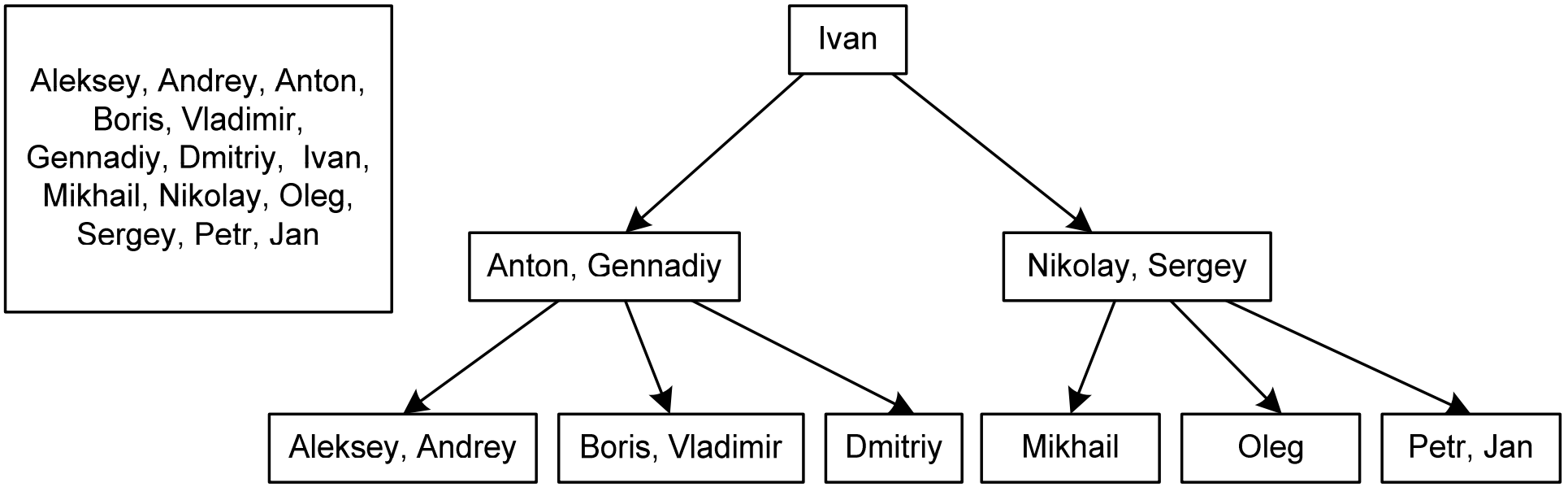
Индекс могут быть реализованы как B - деревья, **B+ - деревья**, хэш-функции

B-деревья

B-дерево порядка n – совокупность иерархически связанных страниц

- Каждая страница содержит от n до 2^n элементов
- Если нелистовая станица содержит m ключей то страница имеет m+1 потомков
- Все листовые страницы находятся на одном уровне

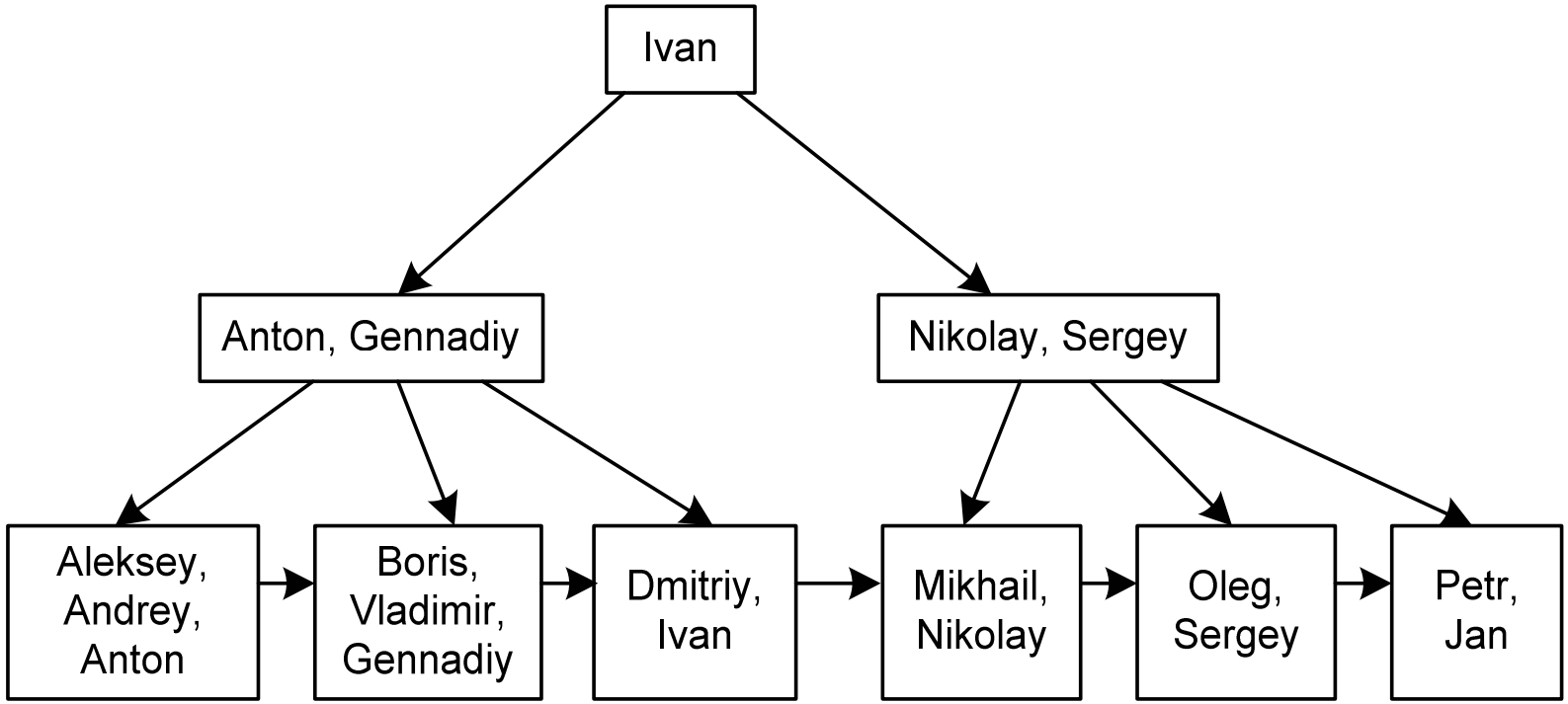
Количество уровней – глубина дерева



B+-деревья

Отличия B+-дерева:

- В нелистовых станицах хранятся только значения ключей (без данных)
- В листовых страницах хранятся значения ключей и данные (ссылки), упорядоченные по ключу



Индексы #2

Создание индексов

CREATE INDEX <index_name> **ON** <table_name> (<column_name> [DESC],…)

Использование индексов

- Выборка по условию
- Сортировка
- Группировка
- Соединение таблиц

Операции добавления и удаления, происходит обновление индексов

- Добавление ключей в станицы
- Расщепление и склеивание страниц

Статистика по индексам (число записей, доля повторений,…)

Выполнение запросов

Перед выполнением запроса происходит его подготовка – составление **плана выполнения запроса** (последовательности обхода и соединения таблиц).

Операции извлечения данных:

NATURAL – полный перебор, до тех пор пока не найдет требуемые данные, допустим при извлечении всех данных таблицы;

INDEX(<index1>,…) – поиск по индексу, обычно более эффективно, используется при соединениях и вычислении условий;

ORDER <index> - полный перебор с упорядочиванием по заданному индексу, можно использовать при **order by** и **group by**.

Выполнение запросов #2

Операции над данными:

JOIN (<select1>,<select2>,…) – соединение двух или более потоков в один, осуществляется перевод всех записей <select1> и поиск для них записей <select2> и т.д., эффективное слияние при наличии индексов;

MERGE (<select1>,<select2>,…) – выбирает и сортирует сразу все потоки и производит слияние за один проход, эффективен при отсутствии индексов

SORT(<select>) – сортировка потока.

При автоматическом создании планов используется статистика по индексам

Примеры планов

SELECT * FROM students;

PLAN (STUDENTS **NATURAL**)

SELECT * FROM students ORDER BY age;

PLAN (STUDENTS **ORDER** AGE_IDX)

SELECT groups.id as gid, count(students.id) AS s_count
FROM groups, students
WHERE students.group_id = groups.id GROUP BY groups.id

PLAN **JOIN** (GROUPS ORDER PK_GROUPS, STUDENTS INDEX (FK_STUD))

5*1000

PLAN **SORT** (JOIN (STUDENTS **NATURAL**, GROUPS INDEX (PK_GROUPS)))

1000*1000

Оптимизация запросов

Перед выполнением запроса происходит его подготовка – составление **плана выполнения запроса** (последовательности обхода и соединения таблиц).

Возможно явное указание плана в запросе – перед **ORDER BY**.

Способы оптимизации выполнения запросов

- Использование индексов
- Использование планов
- Переупорядочивание операторов в запросе
- Использование явной последовательности соединения (JOIN)
- Частичная денормализация БД

Демонстрация

Типичные запросы и способы их оптимизации (IBExpert).