



# Секционирование



Назначение

Табличное наследование

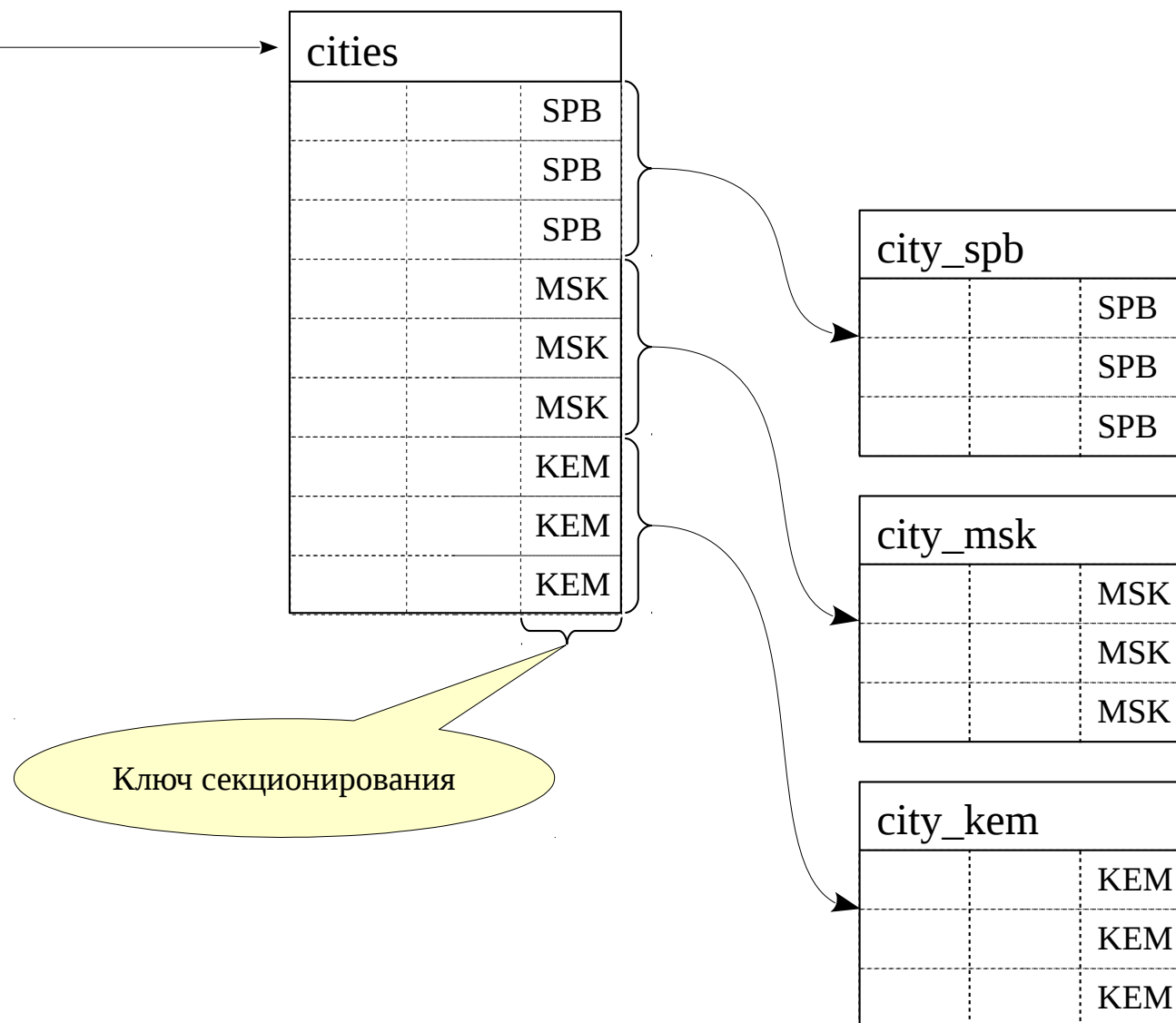
Секционирование через наследование

Типовые задачи

Планы

# Секционирование

SELECT  
INSERT  
UPDATE  
DELETE



## Ускорение запросов

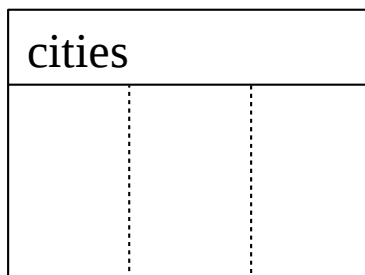
планировщик исключает ненужные секции

## Удобство администрирования

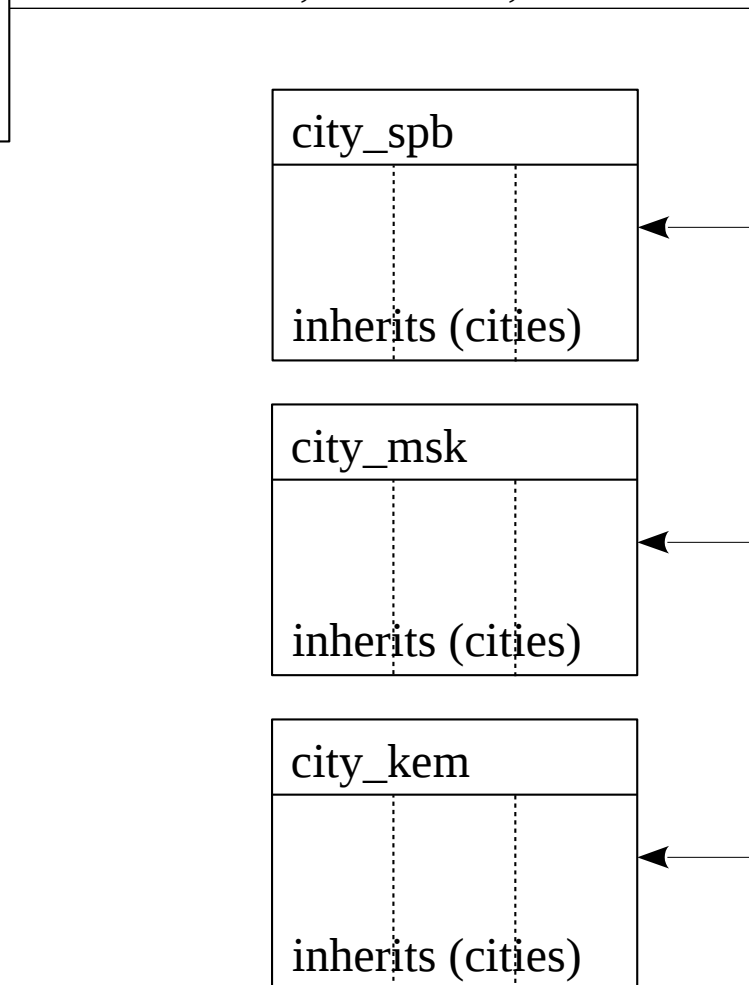
задачи сопровождения: очистка, сбор статистики, переиндексация  
использование разных табличных пространств для секций  
включение/отключение секций

# Табличное наследование

SELECT  
INSERT  
UPDATE  
DELETE

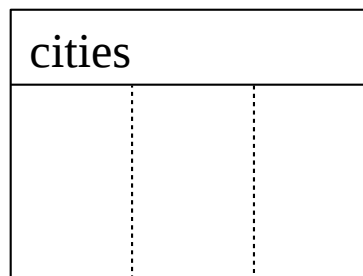


SELECT, UPDATE, DELETE

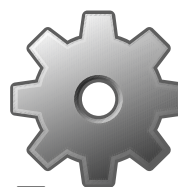


# Вставка записей

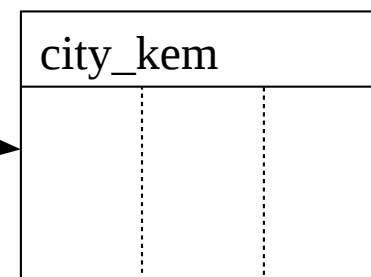
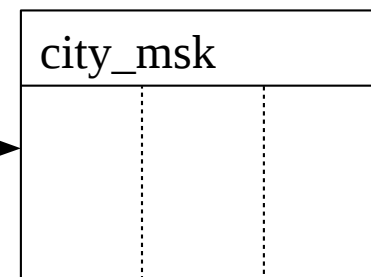
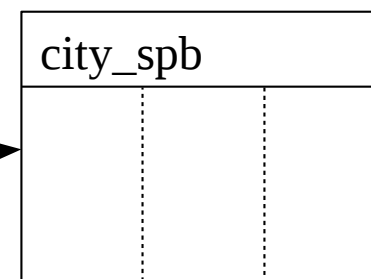
SELECT  
INSERT  
UPDATE  
DELETE



INSERT



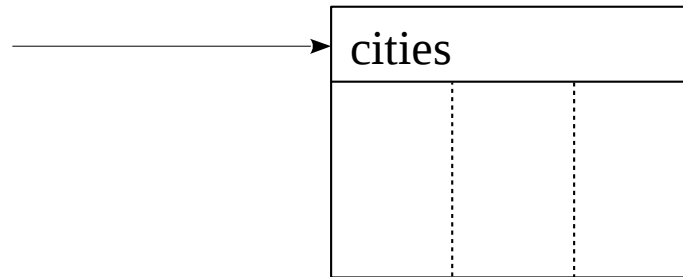
Триггер



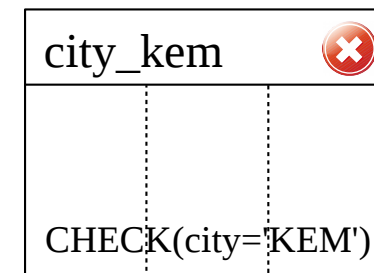
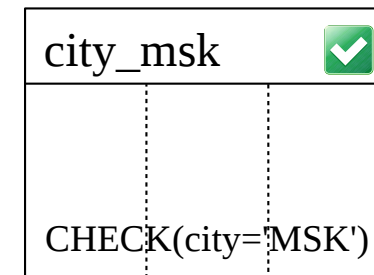
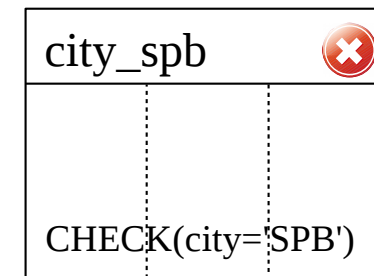
```
if new.city = 'SPB' then
  insert into city_msk ...
elsif new.city = 'MSK' then
  insert into city_spb ...
...
end if;
```

# Исключение секций

```
SELECT  
UPDATE  
DELETE  
...  
WHERE  
  city = 'MSK'
```



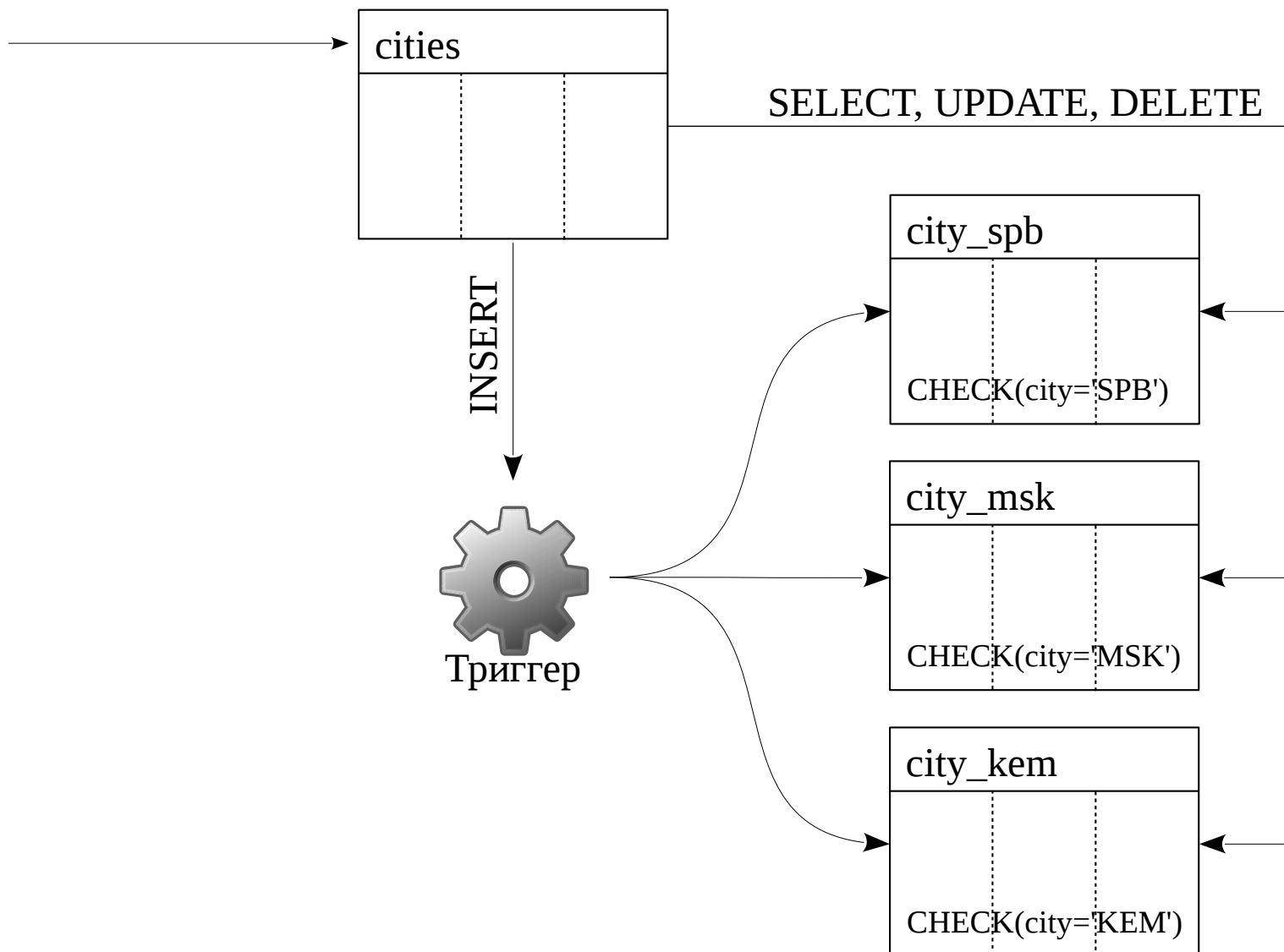
SELECT, UPDATE, DELETE



CONSTRAINT\_EXCLUSION = partition

# Механизм секционирования

SELECT  
INSERT  
UPDATE  
DELETE



Создать новую секционированную таблицу

Добавить новую секцию в таблицу

Отключить или удалить секцию

Секционировать существующую таблицу

Создать родительскую таблицу

Создать дочерние таблицы-секции

Добавить ограничения проверки на секции

Добавить триггер на вставку в родительскую таблицу

Действия при изменении ключа секционирования

# Добавление секции

Создать дочернюю таблицу-секцию

Добавить ограничение проверки на секцию

Обновить триггер на вставку в родительскую таблицу

# Отключить/удалить секцию

Отключение секции — отключение наследования

```
ALTER TABLE ... NO INHERIT
```

Проверить триггер на вставку в родительскую таблицу

Отключенную секцию можно

положить в архив (pg\_dump)

удалить (DROP TABLE)

восстановить (ALTER TABLE ... INHERIT)

## Подготовка

- создать дочерние таблицы-секцию
- добавить ограничения проверки на секции

## Запуск для новых записей

- добавить триггер на вставку в родительскую таблицу

## Перенос исторических данных

- транзакционный перенос строк из родительской таблицы по секциям

Нет глобальных индексов

Нельзя сделать первичный или уникальный индекс

Нет внешних ключей на таблицу

Ручное управление

Не работает INSERT ... RETURNING

Ограничения планировщика

Скорость планирования

# Демонстрация



# Планы по развитию

Встроенная поддержка секционирования

Расширение pg\_pathman

PostgreSQL пока не поддерживает встроенное декларативное секционирование

Механизм секционирования использует

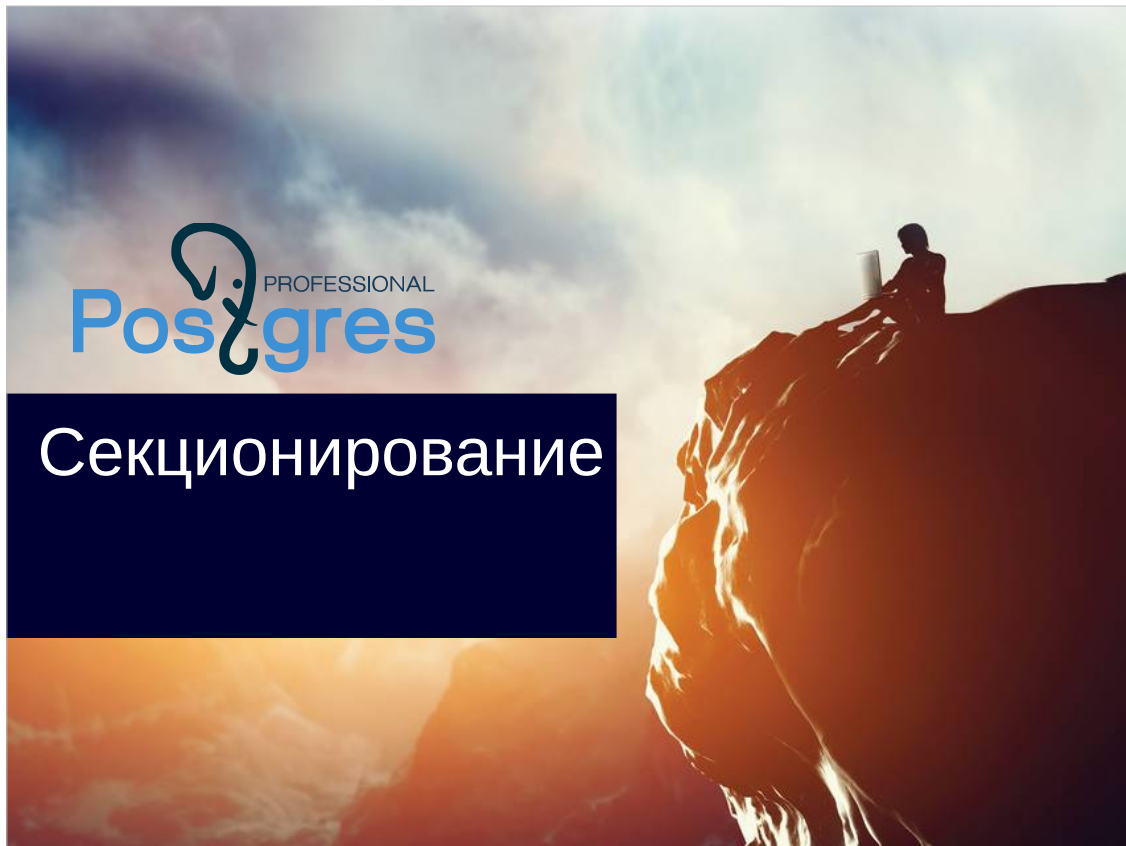
- табличное наследование

- триггер на вставку в родительскую таблицу

- ограничения проверки и `constraint_exclusion`

Такой подход имеет ряд ограничений

1. Создайте базу данных db21.
2. Создайте таблицу dates со столбцом ts типа timestamp.
3. Наполните произвольными данными за один календарный год.
4. Секционировать таблицу dates по кварталам этого года.



### **Авторские права**

Курс «Администрирование PostgreSQL 9.5. Расширенный курс»

© Postgres Professional, 2016 год.

Авторы: Егор Рогов, Павел Лузанов

### **Использование материалов курса**

Некоммерческое использование материалов курса (презентации, демонстрации) разрешается без ограничений. Коммерческое использование возможно только с письменного разрешения компании Postgres Professional. Запрещается внесение изменений в материалы курса.

### **Обратная связь**

Отзывы, замечания и предложения направляйте по адресу:

[edu@postgrespro.ru](mailto:edu@postgrespro.ru)

### **Отказ от ответственности**

Компания Postgres Professional не несет никакой ответственности за любые повреждения и убытки, включая потерю дохода, нанесенные прямым или косвенным, специальным или случайным использованием материалов курса. Компания Postgres Professional не предоставляет каких-либо гарантий на материалы курса. Материалы курса предоставляются на основе принципа «как есть» и компания Postgres Professional не обязана предоставлять сопровождение, поддержку, обновления, расширения и изменения.

Назначение

Табличное наследование

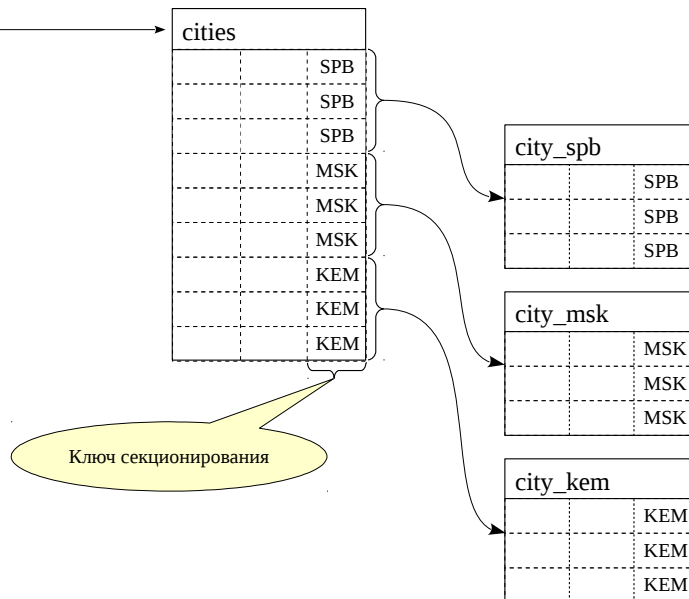
Секционирование через наследование

Типовые задачи

Планы

# Секционирование

SELECT  
INSERT  
UPDATE  
DELETE



Секционирование (Partitioning, англ.) используется в различных СУБД при работе с большими таблицами.

Секционированная таблица представляет собой набор секций, которые разделены в соответствии с ключом секционирования. В примере на картинке, ключ секционирования — это столбец (city), содержащий названия городов.

При этом для приложения все выглядит абсолютно прозрачно. В работе с секционированной таблицей используются обычные команды DML.

## Ускорение запросов

планировщик исключает ненужные секции

## Удобство администрирования

задачи сопровождения: очистка, сбор статистики, переиндексация

использование разных табличных пространств для секций

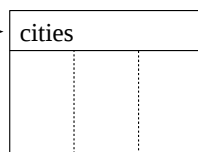
включение/отключение секций

Разделение таблицы на секции служит для решения нескольких задач.

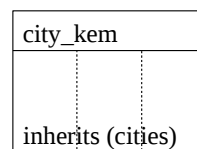
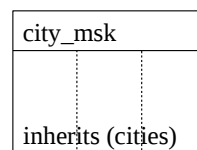
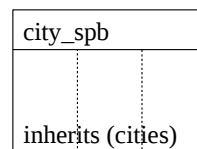
Во-первых, существенно ускоряются некоторые запросы к таблице. Это достигается за счет того, что планировщик, на основании текста запроса, может исключить из плана доступ к отдельным секциям (возможно к большинству).

Во-вторых, управлять отдельными секциями значительно проще, чем одной большой таблицей. Секции можно размещать на разных физических дисках за счет использования табличных пространств. Например, часто используемые секции — на быстрых дисках, редко используемые — на медленных. Неиспользуемые или редко используемые можно вообще отключить (предварительно сделав их копию) и подключить обратно, если возникнет необходимость.

SELECT  
INSERT  
UPDATE  
DELETE



SELECT, UPDATE, DELETE



Функциональность табличного наследования предполагает, что одни таблицы могут наследоваться от других. Для этого при создании дочерней таблицы используется ключевое слово INHERITS.

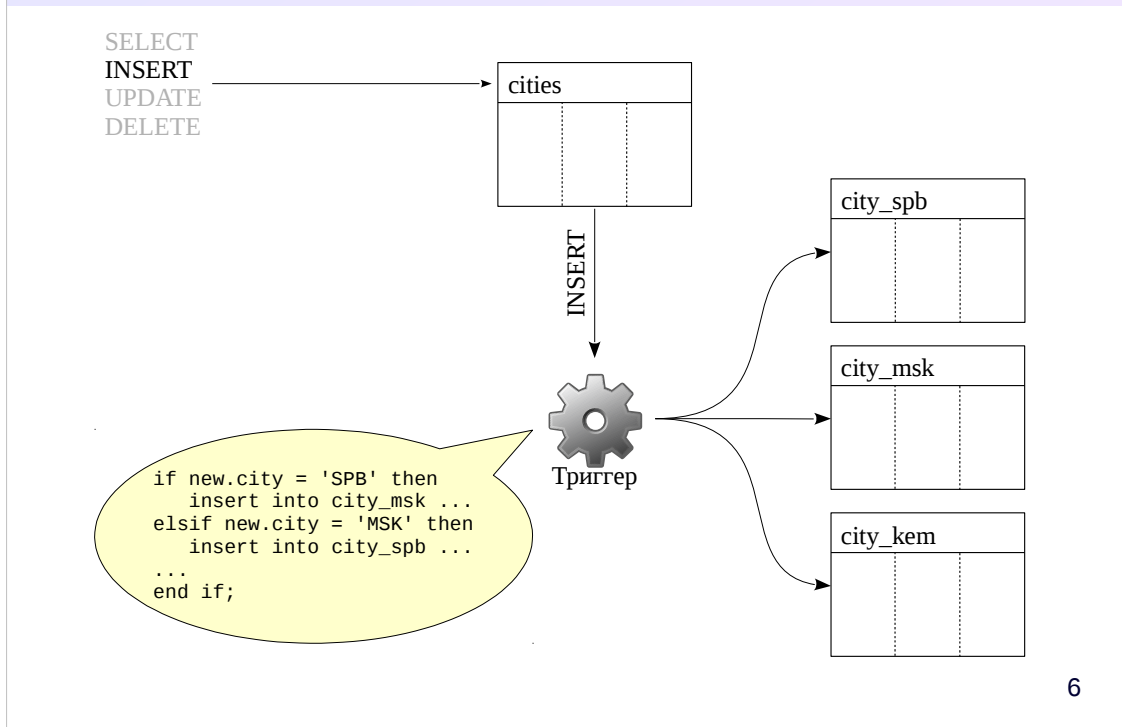
Созданная таким образом таблица автоматически получит все столбцы родительской таблицы. Более того, команды SELECT, UPDATE, DELETE к родительской таблице будут дополнительно выбирать данные и из всех дочерних (если перед родительской таблицей не указано ONLY).

Дочерние таблицы могут иметь дополнительные столбцы, которых нет в родительской. Допускается множественное наследование. Но для целей секционирования важно лишь то, что каждая дочерняя таблица имеет все столбцы родительской. А также то, что запросы к родительской таблице распространяются и на дочерние.

Подробнее о табличном наследовании:

<http://www.postgresql.org/docs/9.5/interactive/ddl-inherit.html>

# Вставка записей



Табличное наследование не распространяется на вставку записей.

Замаскировать распределение записей по секциям можно, создав триггер на вставку в родительскую таблицу. В триггере, вставка записи осуществляется в конкретную дочернюю таблицу, имя которой определяется на основании значения ключа секционирования.

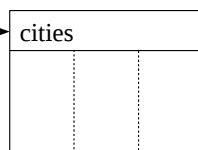
Недостатками этого подхода является замедление вставки за счет дополнительного вызова триггера и невозможность использовать конструкцию `INSERT ... RETURNING`.

Поэтому при массовой вставке записей, если известны названия секций, лучше вставлять записи напрямую в дочерние таблицы.

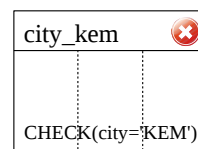
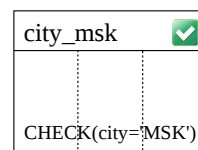
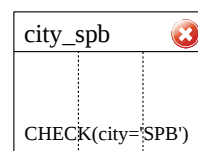
Если допускается изменение ключа секционирования, то дополнительно необходим триггер на операцию `update`. В триггере нужно удалить запись со старым ключом секционирования и вставить запись в таблицу-секцию с новым ключом секционирования.

# Исключение секций

```
SELECT  
UPDATE  
DELETE  
...  
WHERE  
  city = 'MSK'
```



SELECT, UPDATE, DELETE



CONSTRAINT\_EXCLUSION = partition

7

Планировщик сможет исключать обращения к ненужным секциям при выполнении всех следующих условий:

При построении плана уже известны значения ключа секционирования.

Фраза WHERE запроса, как правило, должна содержать константы. Например, нельзя использовать функции, не объявленные как IMMUTABLE.

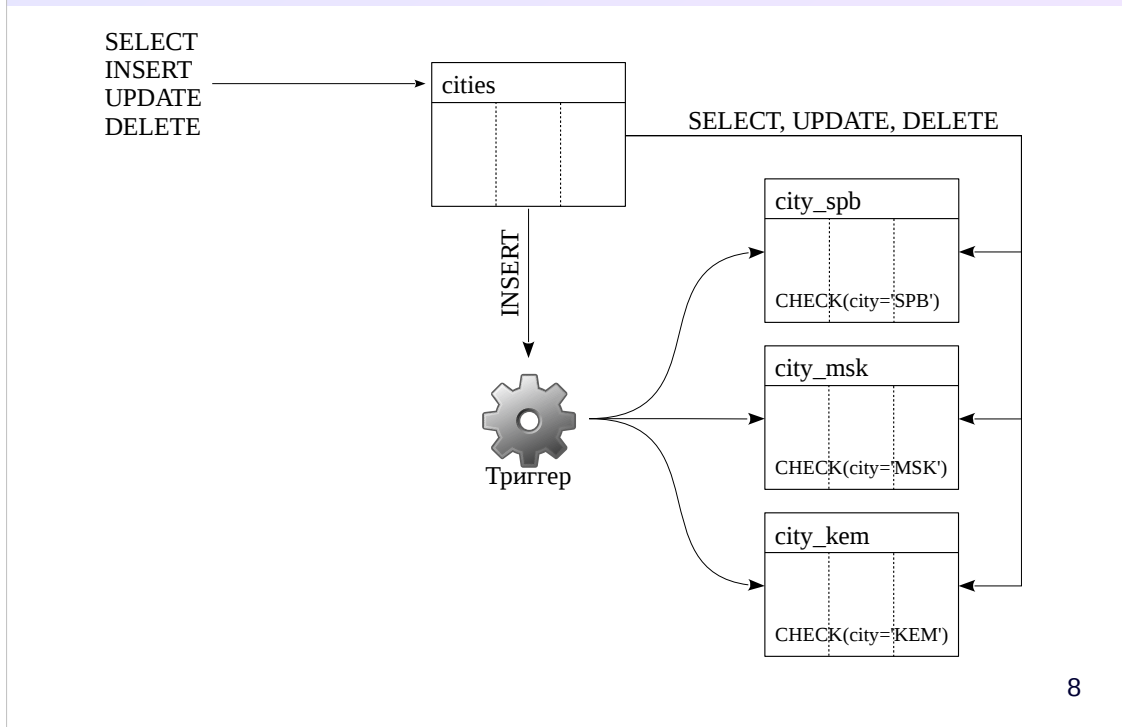
Дочерние таблицы-секции имеют ограничения проверки.

Можно настроить секционирование по списку значений (как в примере), а также по диапазонам значений (например, по периодам времени: дни, месяцы, годы, ...). Диапазоны могут быть неодинакового размера. Обратная сторона: возможны пересечения и «дыры».

Параметр constraint\_exclusion имеет значение 'partition' или 'on'.

'partition' — проверка ограничений для таблиц, участвующих в наследовании, используется по умолчанию.

'on' — проверка ограничений у всех таблиц запроса.



Таким образом, реализация секционирования через табличное наследование заключается в создании:

- Родительской таблицы и таблиц-секций, наследуемых от родительской.
- Триггера на вставку в родительскую таблицу.
- Ограничений проверки на таблицы-секции.

Подробнее о секционировании через табличное наследование:  
<http://www.postgresql.org/docs/9.5/interactive/ddl-partitioning.html>

Создать новую секционированную таблицу

Добавить новую секцию в таблицу

Отключить или удалить секцию

Секционировать существующую таблицу

При работе с секционированием есть ряд типовых задач, таких, как создание новой таблицы, добавление новой секции, удаление/отключение секции, разделение на секции существующей большой таблицы.

Так как процесс секционирования во многом ручной, то для выполнения каждой из этих задач нужно соблюсти определенную последовательность действий.

Далее подробно рассмотрена каждая из этих задач.

Создать родительскую таблицу

Создать дочерние таблицы-секции

Добавить ограничения проверки на секции

Добавить триггер на вставку в родительскую таблицу

Действия при изменении ключа секционирования

Для создания новой секционированной таблицы нужно выполнить:

- Создать родительскую таблицу. Родительскую таблицу, как правило, оставляют пустой, но это необязательно.
- Создать дочерние таблицы.
- Добавить ограничения проверки на каждую дочернюю таблицу, в соответствии с ключем секционирования.
- Добавить триггер на вставку записей в родительскую таблицу.
- Дополнительно нужно убедиться, что изменение ключа секционирования невозможно. Иначе требуется триггер на команду update, переключив запись из одной секции в другую.

Изменение ключа секционирования обычно недопустимо, поэтому в дальнейшем не будет упоминаний про триггер на update.

Создать дочернюю таблицу-секцию

Добавить ограничение проверки на секцию

Обновить триггер на вставку в родительскую таблицу

Если секции заканчиваются, например, секционирование по периодам времени, то периодически нужно добавлять новые. Для этого:

- Создать дочернюю таблицу (таблицы).
- Добавить ограничение проверки на новые таблицы.
- Обновить триггер на вставку в родительскую таблицу, так чтобы он теперь вставлял записи в новые секции.

Отключение секции — отключение наследования

`ALTER TABLE ... NO INHERIT`

Проверить триггер на вставку в родительскую таблицу

Отключенную секцию можно

положить в архив (`pg_dump`)

удалить (`DROP TABLE`)

восстановить (`ALTER TABLE ... INHERIT`)

Для отключения секции достаточно:

- Отключить наследование.
- Проверить триггер на вставку. Возможно из него нужно убрать ту часть, которая вставляет строки в отключенные секции.

После отключения секции:

- Можно сделать ее копию, например командой `COPY` или утилитой `pg_dump`.
- Удалить целиком командой `DROP TABLE`.

Впоследствии можно повторно подключить секцию. Для этого:

- Восстановить таблицу из копии.
- Проверить ограничение проверки.
- Включить наследование командой `ALTER TABLE ... INHERIT`.
- Проверить триггер на вставку.

## Подготовка

- создать дочерние таблицы-секцию
- добавить ограничения проверки на секции

## Запуск для новых записей

- добавить триггер на вставку в родительскую таблицу

## Перенос исторических данных

- транзакционный перенос строк из родительской таблицы по секциям

Если таблица выросла в размерах, то ее можно секционировать на лету, не прерывая работу системы. Это делается в три этапа.

### I. Подготовка.

Существующая таблица будет родительской. Поэтому создаем дочерние таблицы-секции. Добавляем ограничения проверки на дочерние таблицы. После этого запросы к родительской таблице уже будут обращаться к дочерним. Но в них пока ничего нет.

### II. Запуск для новых записей.

Добавляем триггер на вставку в родительскую таблицу. После этого все новые записи уже будут распределяться по новым секциям. Но родительская таблица по-прежнему будет содержать все накопленные данные.

### III. Перенос исторических данных из родительской таблицы.

Теперь можно приступить к переносу данных из родительской таблицы в дочерние. Нужно удалить данные из родительской таблицы и вставить их в нужные секции. Важно выполнять операции удаления и вставки в одной транзакции, чтобы параллельно работающие запросы ничего не «потеряли».

Например, можно использовать команду:

```
with (delete ... returning ...) insert ...
```

В зависимости от количества строк, перенос может занять много времени. Но его можно выполнять постепенно, не прерывая работы приложения.

Нет глобальных индексов

Нельзя сделать первичный или уникальный индекс

Нет внешних ключей на таблицу

Ручное управление

Не работает INSERT ... RETURNING

Ограничения планировщика

Скорость планирования

Секционирование на табличном наследовании имеет ряд ограничений. Многие из них основаны на том факте, что секции — это по сути отдельные таблицы:

Нельзя создать глобальный индекс по всей секционированной таблице.

Нельзя создавать первичный и уникальные ключи.

Нельзя ссылаться на секционированную таблицу.

Кроме того:

Ручное управление секционированием (начальное создание, добавление новых секций, и т. д.). Потенциальный источник ошибок.

Не работает конструкция INSERT ... RETURNING.

Планировщик уже при построении плана запроса должен знать значения для ключа секционирования. Поэтому фраза WHERE запроса, как правило, должна содержать константы. К примеру, нельзя использовать функции, не объявленные как IMMUTABLE.

С увеличением количества секций падает скорость построения плана запроса, т. к. планировщику требуется проверять ограничения на все большем количестве таблиц.



Встроенная поддержка секционирования

Расширение `pg_pathman`

В сообществе разработчиков активно ведутся работы по встроенной поддержке секционирования.

Компания «Постгрес Профессиональный» выпустила бета-версию расширения `pg_pathman`:

[https://github.com/postgrespro/pg\\_pathman](https://github.com/postgrespro/pg_pathman)

PostgreSQL пока не поддерживает встроенное декларативное секционирование

Механизм секционирования использует

- табличное наследование

- триггер на вставку в родительскую таблицу

- ограничения проверки и `constraint_exclusion`

Такой подход имеет ряд ограничений

1. Создайте базу данных db21.
2. Создайте таблицу dates со столбцом ts типа timestamp.
3. Наполните произвольными данными за один календарный год.
4. Секционировать таблицу dates по кварталам этого года.

Для создания и наполнения таблицы можно использовать команды:

```
create table dates (ts timestamp);
insert into dates
  select t.ts from generate_series (
    '2016-01-01'::timestamp
    , '2016-12-31'::timestamp
    , '1 min'::interval
  ) as t (ts);
```

Таблица будет наполнена строками с минутным интервалом за все дни 2016 года.