Индексы

Определение

- **Индекс.** Структура данных, которая помогает СУБД быстрее обнаружить отдельные записи а файле и сократить время выполнения запросов пользователей

Типы индексов

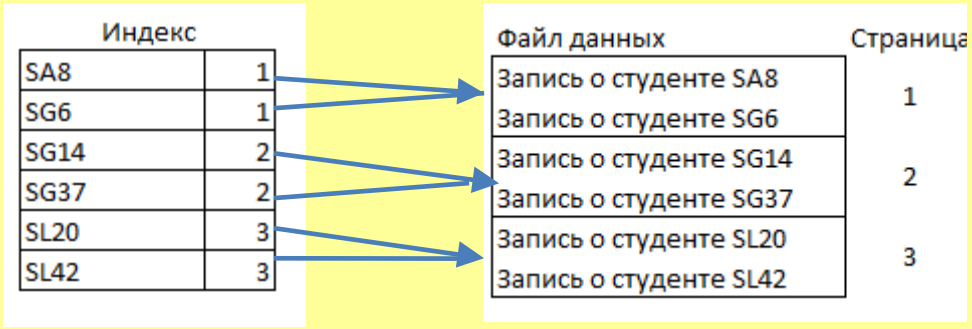
- **Первичный индекс**. Файл данных последовательно упорядочивается по полю ключа упорядочения , а на основе поля ключа упорядочения создается поле индексации, которое гарантированно имеет уникальное значение в каждой записи.
- **Индекс кластеризации**. Файл данных последовательно упорядочивается по неключевому полю, и на основе этого неключевого поля формируется поле индексации, поэтому в файле может быть несколько записей, соответствующих значению этого поля индексации.
  Неключевое поле называется *атрибутом кластеризации. (Кластерный)*
- ***Вторичный индекс***. Индекс, который определен на поле файла данных, отличном от поля, по которому выполняется упорядочение. (Некластерный)
- Файл может иметь не больше одного первичного индекса или одного индекса кластеризации, но дополнительно к ним может иметь несколько вторичных индексов.

индексированный последовательный файл

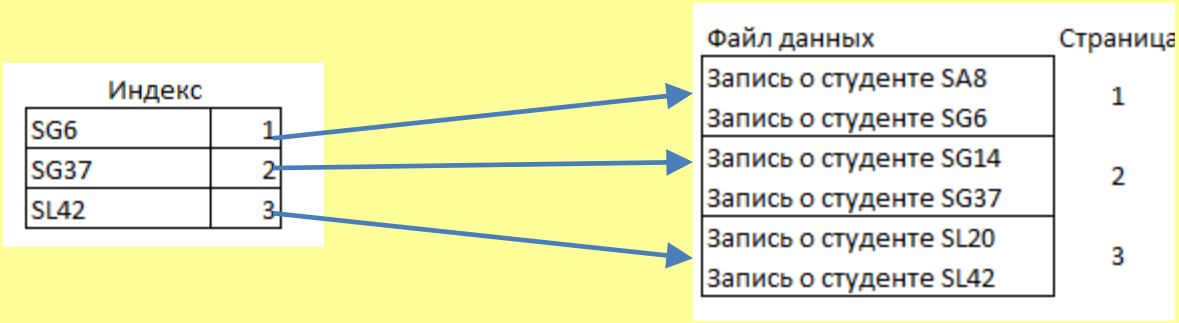
- Отсортированный файл данных с первичным индексом называется *индексированным последовательным файлом,* или индексно-последовательным файлом.Эта структура является компромиссом между файлами с полностью последовательной и полностью произвольной организацией. В таком файле записи могут обрабатываться как последовательно, так и выборочно, с произвольным доступом, осуществляемым на основу поиска по заданному значению ключа с использованием индекса. Индексированный последовательный файл имеет более универсальную структуру, которая обычно включает следующие компоненты:
- • первичная область хранения;
- • отдельный индекс или несколько индексов;
- • область переполнения. (Область расширения)
- Подобная организация файлов используется в методе индексно - последовательного доступа (Indexed Sequential Access Method — ISAM)

Плотный и разреженный индекс

- Плотный индекс



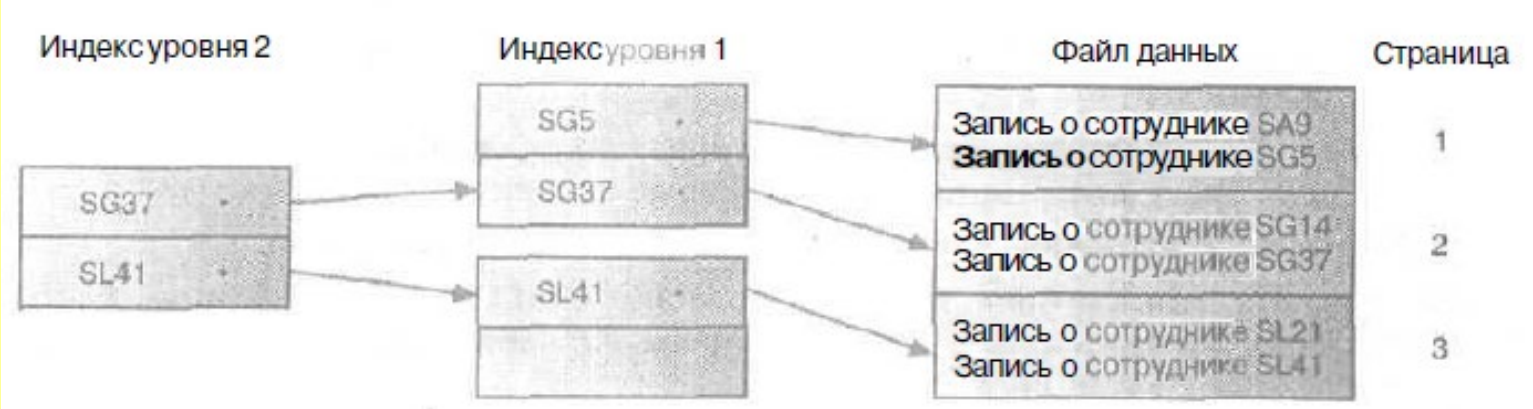
- Разреженный индекс



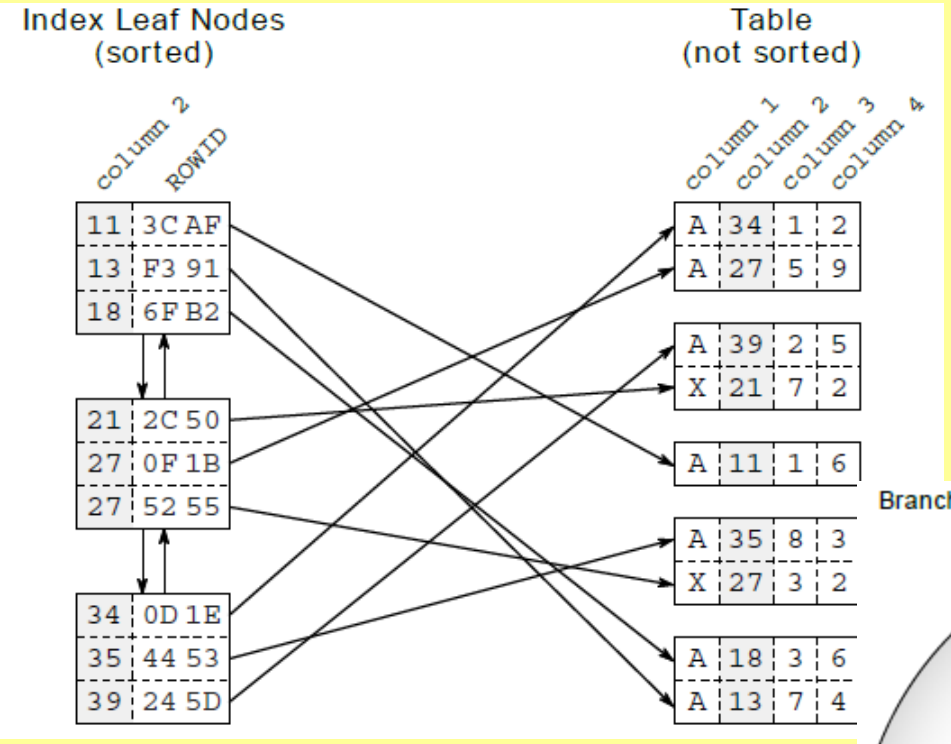
Вторичный индекс

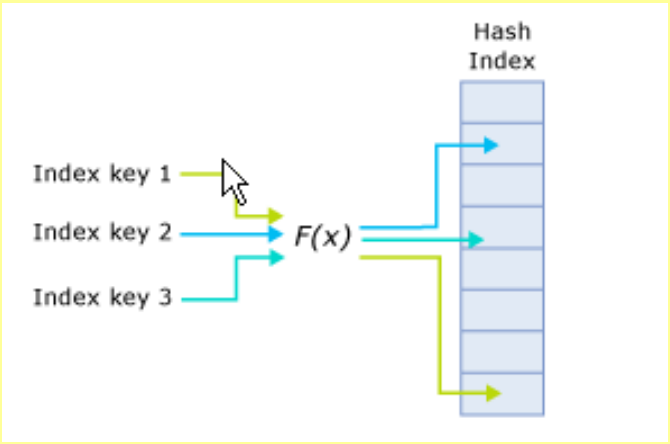
- Вторичный индекс также является упорядоченным файлом, аналогичным первичному индексу. Однако связанный с первичным индексом файл данных всегда отсортирован по ключу этого индекса, тогда как файл данных, связанный со вторичным индексом, не обязательно должен быть отсортирован по ключу индексации. Кроме того, ключ вторичного индекса может содержать повторяющиеся значения, что не допускается для значений ключа первичного индекса.

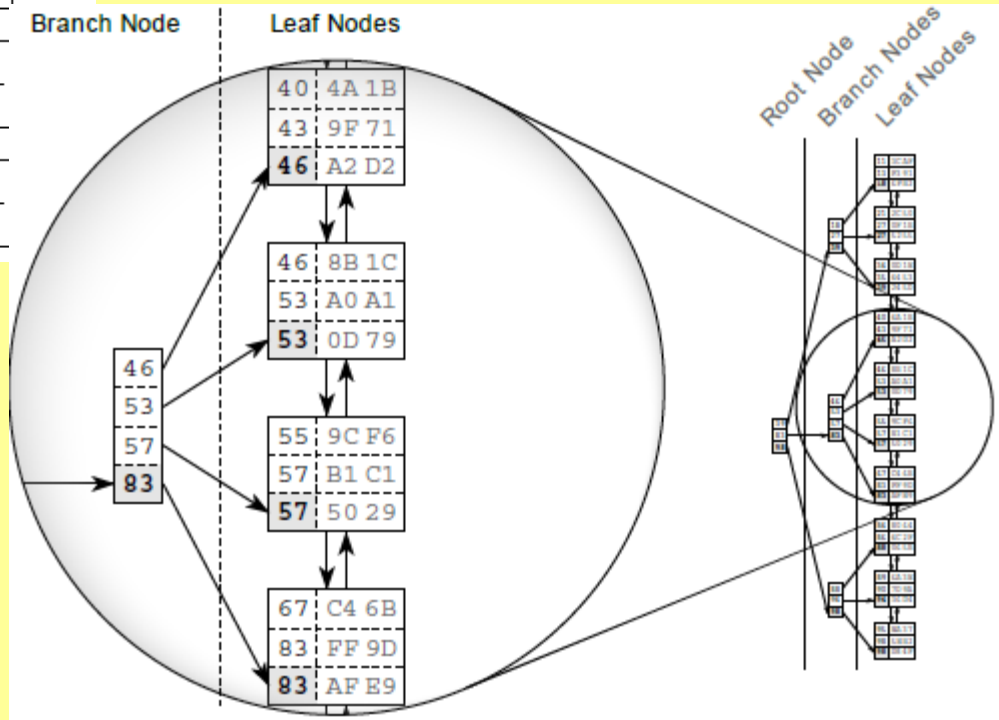
Многоуровневые индексы



Типы индексов по организации хранения







Пример вставки в B*-дерево

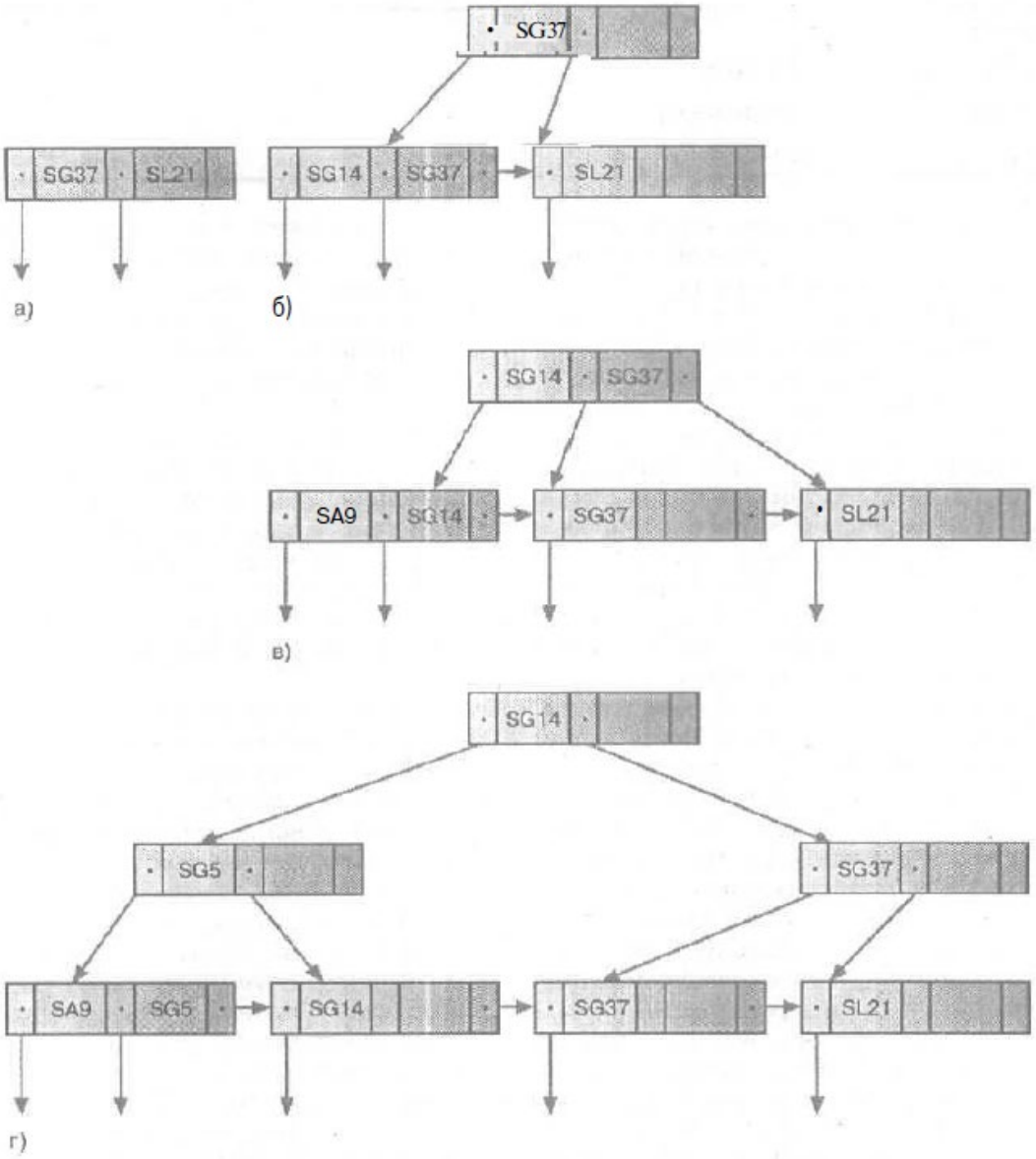


Рис. В.10. Пример вставки записей в B+-дерево: а) после вставки записей SL21 и SG37; б) после вставки записи SG14; в) после вставки записи SA9; г) после вставки записи SG5

Index Organized Tables (IOT) in Oracle

- SQL Server и MySQL (с использованием InnoDB) имеют более широкое представление о том, что «Индекс» означает. Относят это понятие к таблицам, которые состоят из структуры индекса только в виде кластерных индексов. Эти таблицы в базе данных Oracle называются организованными по индексу таблицы (IOT index organized tables)
- В индексных организованных таблицах (IOT) данные первичного ключа и столбца неключевого поля хранятся в одной и той же древовидной структуре (B * Tree). Фактически данные хранятся в индексе первичного ключа.

Создание синтаксис

- CREATE [UNIQUE | FULLTEXT | SPATIAL] INDEX *index_name* [*index_type*] ON *tbl_name* (*key_part*,...) [*index_option*] [*algorithm_option* | *lock_option*] ...
- *key_part*: {*col_name* [(*length*)] | (*expr*)} [ASC | DESC]
- *index_option*: KEY_BLOCK_SIZE [=] *value* | *index_type* | WITH PARSER *parser_name* | COMMENT '*string*' | {VISIBLE | INVISIBLE}
- *index_type*: USING {BTREE | HASH}
- *algorithm_option*: ALGORITHM [=] {DEFAULT | INPLACE | COPY}
- *lock_option*: LOCK [=] {DEFAULT | NONE | SHARED | EXCLUSIVE}

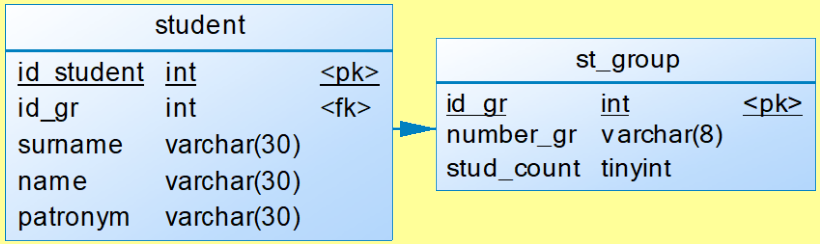
Алгоритмы (действия над таблицей)

- COPY: Операции выполняются с копией исходной таблицы, и данные таблицы копируются из исходной таблицы в новую строку строка за строкой. Параллельное выполнение команд DML не допускается.
- INPLACE: Операции избегают копирования данных таблицы, но могут перестроить таблицу на месте. Исключительная блокировка метаданных на столе может быть кратко взята на этапах подготовки и выполнения операции. Как правило, Параллельное выполнение команд DML поддерживается.

Разрешенные типы индексов

Storage Engine	Разрешенные типы индексов
InnoDB	BTREE
MyISAM	BTREE
MEMORY/HEAP	HASH, BTREE
NDB	HASH, BTREE

Пример



- CREATE INDEX surname_ind on student (surname);
- create index surname_ind2 on student (surname(10));

Функциональные ключевые части в индексе с (8.0.13)

- CREATE TABLE t1
  (col1 INT,
  col2 INT,
  INDEX func_index ((ABS(col1))));
- CREATE INDEX idx1 ON t1 ((col1 + col2)); CREATE INDEX idx2 ON t1 ((col1 + col2), (col1 - col2), col1);
- ALTER TABLE t1 ADD INDEX ((col1 * 40) DESC);

Когда создавать индексы?

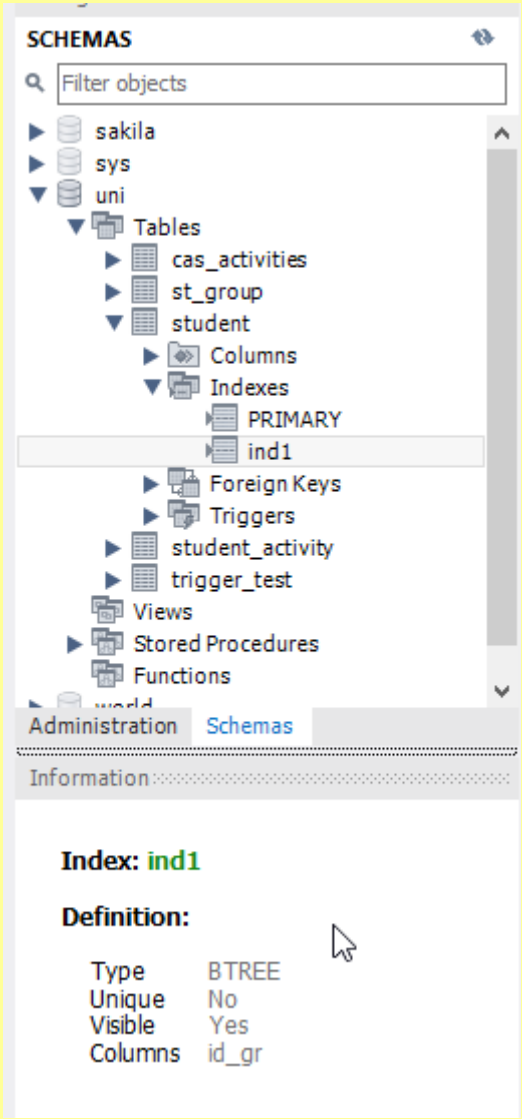
- Индексы следует **создавать по мере обнаружения медленных запросов**. В slow log в MySQL. Запросы, которые выполняются более 1 секунды являются первыми кандидатами на оптимизацию.
- Начинайте создание индексов с **самых частых запросов**. Запрос, выполняющийся секунду, но 1000 раз в день наносит больше ущерба, чем 10-секундный запрос, который выполняется несколько раз в день.
- **Не создавайте индексы** на таблицах, число записей в которых **меньше нескольких тысяч**. Для таких размеров выигрыш от использования индекса будет почти незаметен.
- **Не создавайте** индексы **заранее**, например, в среде разработки. Индексы должны устанавливаться исключительно под форму и тип нагрузки работающей системы.
- Удаляйте неиспользуемые индексы.

Куда ставить индексы

- Индексы создаются на поля, по которым происходит отбор данных в запросе,
- Индексы создаются на поля, по которым происходит группировка в запросе
- Индексы создаются на поля, по которым связываются таблицы

- **Не создаются** индексы на поля **с малым разнообразием значений** (пол человека)

Как узнать какие индексы есть в базе (workbench)



Как узнать какие индексы есть в базе

- SELECT DISTINCT TABLE_NAME, INDEX_NAMEFROM INFORMATION_SCHEMA.STATISTICSWHERE TABLE_SCHEMA = 'database_name';
- SELECT DISTINCT TABLE_NAME, INDEX_NAMEFROM INFORMATION_SCHEMA.STATISTICSWHERE TABLE_SCHEMA = 'database_name' and TABLE_NAME=‘table_name';

Как узнать какие индексы есть в базе

- SELECT DISTINCT TABLE_NAME, INDEX_NAMEFROM INFORMATION_SCHEMA.STATISTICSWHERE TABLE_SCHEMA = 'uni';
- SELECT DISTINCT TABLE_NAME, INDEX_NAMEFROM INFORMATION_SCHEMA.STATISTICSWHERE TABLE_SCHEMA = 'uni' and TABLE_NAME='student_activity';

TABLE_NAME	INDEX_NAME
cas_activities	PRIMARY
st_group	PRIMARY
student	ind1
student	PRIMARY
student_activity	FK_Relationship_2
student_activity	PRIMARY

TABLE_NAME	INDEX_NAME
student_activity	FK_Relationship_2
student_activity	PRIMARY

Как посмотреть индексы более детально

- SHOW INDEX FROM table_name FROM db_name;
- SHOW INDEX FROM db_name.table_name;

Формат сведений об индексах

Столбец	Описание
Table	Название таблицы.
Non_unique	0, если индекс не может содержать дубликаты, 1, если это возможно.
Key_name	Название индекса. Если индекс является первичным ключом, имя всегда PRIMARY.
Seq_in_index	Порядковый номер столбца в индексе, начиная с 1.
Column_name	Название столбца.
Collation	Как столбец сортируется в индексе. Это может иметь значения A (по возрастанию), D (по убыванию) или NULL (не отсортировано).
Cardinality	Оценка количества уникальных значений в индексе. Чтобы обновить этот номер, запустите ANALYZE TABLE или (для таблиц MyISAM) myisamchk -a.
Sub_part	Префикс индекса. Количество проиндексированных символов, если столбец только частично проиндексирован, NULL, если весь столбец проиндексирован. (в байтах)
Packed	Указывает, как сжат ключ. NULL, если это не так.
Null	Содержит YES, если столбец может содержать значения NULL, и '', если нет.
Index_type	Используемый метод индекса (BTREE, FULLTEXT, HASH, RTREE).
Comment	Информация об индексе не описанная в его собственном столбце, например disabled, если индекс отключен.
Index_comment	Комментарий для индекса ,выставленный в COMMENT при создании индекса.
Visible	Виден ли индекс оптимизатору.
Expression	MySQL 8.0.13 и выше поддерживает использование функциональной ключевой части, которые влияют на столбцы Column_name и Expression: •Для нефункциональной ключевой части Column_name указывает столбец, индексированный ключевой частью, а Expression - NULL. •Для функциональной ключевой части столбец Column_name имеет значение NULL, а Expression указывает выражение для ключевой части.

Как посмотреть индексы более детально

- SHOW INDEX FROM student FROM uni;
- SHOW INDEX FROM uni.student;

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expres
student	0	PRIMARY	1	id_st	A	7	NULL	NULL		BTREE			YES	NULL
student	1	ind1	1	id_gr	A	4	NULL	NULL	YES	BTREE			YES	NULL
student	1	surname_ind	1	surname	A	6	NULL	NULL	YES	BTREE			YES	NULL
student	1	surname_ind2	1	surname	A	6	10	NULL	YES	BTREE			YES	NULL

Удаление синтаксис

- DROP INDEX *index_name* ON *tbl_name* [*algorithm_option* | *lock_option*] ...

algorithm_option: ALGORITHM [=] {DEFAULT | INPLACE | COPY}

lock_option: LOCK [=] {DEFAULT | NONE | SHARED | EXCLUSIVE}

Удаление пример

- DROP INDEX surname_ind2 on student