??????????, ??????? ? ???????????

Транзакции

Блокировки. Записывающие должны запрашивать и получать блокировку записи для изменения данных, а извлекающие должны запрашивать и получать блокировку чтения. Варианты блокировки:

- запросы на чтение блокируются пока блокировка записи не снята, за один раз для каждой таблицы (или ее части) только одна блокировка записи
- блокировка записи, не нужны блокировки чтения. Но сервер гарантирует, что каждый читатель видит согласованное представление данных до завершения запроса (versioning).

Уровни блокировок

- Блокировка таблиц Не позволяет нескольким пользователям одновременно изменять данные в одной таблице.
- Блокировка страниц Не позволяет нескольким пользователям изменять данные в одной и той же странице (страница — это сегмент памяти, обычно в диапазоне от 2 до 16 Кбайт) таблицы одновременно.
- Блокировка строк Не позволяет нескольким пользователям одновременно изменять одну и ту же строку в таблице.

Есть режим автофиксации и транзакции.

- По умолчанию одиночная инструкция INSERT, UPDATE или DELETE неявно включается в транзакцию и немедленно подтверждается.
- SET AUTOCOMMIT=0 отключает для сессии
- только для транзакционных таблиц
- Некоторые команды заставляют подтвердить транзакцию до их выполнения. Команды (Data Definition Language, DDL) типа ALTER TABLE, LOCK TABLES, ...

start transaction начало транзакации, commit - завершение, rollback - возврат.

Механизмы хранения mysql:

MyISAM	Нетранзакционный, табличная блокировка, скоростной доступ на чтение с малым кол-вом записи

MEMORY	Нетранзакционный, табличная блокировка, для скоростного кэша, таблицы в памяти, при нехватке своп на диск, не поддерживает TEXT BLOB
CSV	Транзакционный, данные в файлах с данными с разделением запятыми, обмен данными. Нет индексов.
InnoDB	Транзакционный, блокировка на уровне строки
Merge	Специальный механизм, используемый для создания нескольких идентичных таблиц MyISAM в виде единой таблицы (так называемое разбиение таблицы)
Archive	хранение больших количеств неиндексированных данных, в основном для архивных целей. Не поддерживает DELETE, UPDATE.
Blackhole	для репликации
federated	Одна база на нескольких серверах. Создает клиентское соединение и выполняет еще раз запрос, получает данные. По-умолчанию выключен.
NDB cluster	

статус таблицы

```
show table status like ’customer' \G;
```

изменить механизм хранения определенной таблицы

```
alter table customer engine = InnoDB;
```

Нельзя сочетать разные механизмы в одной транзакции. управление конкурентным доступом с помощью многоверсионности (multiversion concurrency control, MVCC)

Точки сохранения: Возможность частично откатывать транзакцию.

```
SAVEPOINT my_savepoint;
ROLLBACK TO SAVEPOINT my_savepoint;

START TRANSACTION;
UPDATE product
SET date_retired = CURRENT_TIMESTAMP()
WHERE product_cd = 'XYZ';
SAVEPOINT before close accounts;
UPDATE account
```

```
SET status = 'CLOSED', close_date = CURRENT_TIMESTAMP(),
last_activity_date = CURRENT_TIMESTAMP{)
WHERE product_cd = 'XYZ';
ROLLBACK TO SAVEPOINT before_close_accounts;
COMMIT;
```

Индексы

Индексы независимые объекты, хотя и относятся к базе.

Название	Назначение и создание
Простой индекс	ускоряет поиск данных `ALTER TABLE customer ADD INDEX idx email (email);`
Уникальный индекс	запрещает два элемента с одинаковым значением `ALTER TABLE customer ADD UNIQUE idx_email (email);`
Многостолбцовый индекс	важна последовательность - эффекта не будет при поиске только по последнему столбцу `ALTER TABLE customer ADD INDEX idx_full_name (last_name, first_name);`

Менее 64 индексов на таблицу, 16 столбцов максимум для мультииндекса

Просмотр индексов:

```
SHOW INDEX FROM customer \G;
```

Удаление индекса:

```
ALTER TABLE customer DROP INDEX idx_email;
```

Типы индексов:

- B-Tree Индекс на основе деления отрезка пополам. Хорош для разнородных почти сбалансированных данных

- Битовые маски - когда значений относительно мало. Генерит несколько таблиц, соотв. каждому значению ключа.

```
CREATE BITMAP INDEX idx_active ON customer (active);
```

- Текстовые индексы

Просмотр плана выполнения запроса:

```
EXPLAIN
SELECT customer_id, first_name, last_name
FROM customer
WHERE first_name LIKE 'S%' AND last name LIKE 'P%' \G;
```

Ограничение

Ограничение — условия, накладываемые столбцы таблицы. Типы ограничений:

- Ограничения первичного ключа Определяют столбец или столбцы, для которых гарантируется их уникальность в таблице
- Ограничения внешнего ключа содержать только значения, найденные в столбце первичного ключа другой таблицы (могут также ограничиваться допустимые значения в других таблицах при установке правил update cascade и/или delete cascade)
- Ограничения уникальности Ограничивают один или несколько столбцов таким образом, чтобы они содержали уникальные значения в таблице (ограничение первичного ключа — частный случай ограничения уникальности)
- Проверочные ограничения Ограничивают допустимые значения для столбца

Создание ограничений

при создании таблицы

```
CREATE TABLE customer
customer_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT, ... store_id TINYINT UNSIGNED NOT NULL,
address_id SMALLINT UNSIGNED NOT NULL,
PRIMARY KEY (customer_id), CONSTRAINT fk_customer_address FOREIGN KEY (address_id) REFERENCES
address (address_id) ON DELETE RESTRICT ON UPDATE CASCADE,
CONSTRAINT fk_customer_store FOREIGN KEY (store_id) REFERENCES store (store_id) ON DELETE
RESTRICT ON UPDATE CASCADE
```

при изменении таблицы

```
ALTER TABLE customer ADD CONSTRAINT fk_customer_address FOREIGN KEY (address_id) REFERENCES address (address id) ON DELETE RESTRICT ON UPDATE CASCADE;
```

удаление ограничений

```
ALTER TABLE customer drop
```

Revision #1
Created 7 April 2025 06:53:07 by Admin
Updated 7 April 2025 07:15:56 by Admin