При создании или изменении таблицы вы можете указать следующие свойства столбцов.

• ***NOT NULL***.

Это свойство указывает, что в данном столбце не допускаются неопределенные значе-

ния (NULL).

Если для столбца задано свойство NOT NULL, то, в частности, NULL не может исполь-

зоваться в качестве значения по умолчанию для этого столбца. Значение по умолчанию,

отличное от NULL, вы можете задать с помощью свойства DEFAULT <Значение>, которое

описано ниже. Если же вы задали для столбца свойство NOT NULL, но не задали значение

по умолчанию и не указали значение для этого столбца при вставке строки в таблицу, то

поведение программы MySQL зависит от того, в каком режиме вы работаете (об этом будет

подробно рассказано в подразделе «Вставка отдельных строк»).

• ***NULL***.

Данное свойство указывает, что в столбце разрешены неопределенные значения

(NULL). Задавать это свойство имеет смысл только для столбцов с типом TIMESTAMP,

которые по умолчанию не допускают неопределенных значений. Остальные типы столбцов

допускают неопределенные значения, если только для них не задано свойство NOT NULL.

• ***DEFAULT*** <Значение>.

Данное свойство определяет значение по умолчанию для столбца, которое исполь-

зуется, если при вставке строки в таблицу значение столбца не задано явно. Значением

по умолчанию может быть только константа; исключение составляют столбцы с типом

TIMESTAMP, для которых в качестве значения по умолчанию можно задать переменную

величину CURRENT\_TIMESTAMP (текущую дату и время). Нельзя установить значение по

умолчанию для столбцов с типом TINYBLOB, TINYTEXT, BLOB, TEXT, MEDIUMBLOB,

MEDIUM-TEXT, LONGBLOB и LONGTEXT, а также для числовых столбцов, для которых

задано свойство AUTO\_INCREMENT. Кроме того, нельзя использовать неопределенное зна-

чение по умолчанию (NULL), если для столбца задано свойство NOT NULL.

• ***COMMENT*** 'Текст комментария'.

Произвольное текстовое описание столбца длиной до 255 символов. Например, описа-

ние для поля rating (рейтинг) таблицы Customers (Клиенты) можно задать следующим обра-

зом:

*rating* ***INT******COMMENT*** *'Рейтинг клиента'*

Помимо перечисленных свойств, для столбца можно также задать свойства UNIQUE и

PRIMARY KEY, однако соответствующие настройки ключевых столбцов и индексов можно

указать и после определения всех столбцов таблицы.

**Ключевые столбцы и индексы**

После того как определены все столбцы таблицы, можно перечислить через запятую

ключевые столбцы и индексы (см. листинги 2.1–2.4). Вы можете использовать следующие

конструкции:

• [***CONSTRAINT*** <Имя ключа>] ***PRIMARY KEY*** (<Список столбцов>).

Определяет первичный ключ таблицы (о первичных ключах было рассказано в главе

1). В таблице может быть только один первичный ключ, состоящий из одного или несколь-

ких столбцов. Столбцам, входящим в первичный ключ, автоматически присваивается свой-

ство NOT NULL. Ключевое слово CONSTRAINT и имя ключа можно опустить, так как для

первичного ключа заданное имя игнорируется и используется имя PRIMARY.

• ***INDEX*** [<Имя индекса>] (<Список столбцов>).

Создает индекс для указанных столбцов. Индекс – это вспомогательный объект, позво-

ляющий значительно повысить производительность запросов с условием на значение столб-

цов, включенных в индекс (подробнее об индексах мы поговорим в главе 6).

Имя индекса указывать не обязательно. Если вы не зададите имя индекса, оно сгене-

рируется автоматически.

Вместо ключевого слова INDEX можно использовать его синоним – слово KEY.

• ***FULLTEXT*** [<Имя индекса>] (<Список столбцов>).

Создает полнотекстовый индекс для указанных столбцов. Полнотекстовый индекс

обеспечивает ускоренный поиск по значениям символьных столбцов (типы CHAR,

VARCHAR, TINYTEXT, TEXT, MEDIUMTEXT и LONGTEXT) независимо от длины зна-

чений. Такой индекс подобен предметному указателю в книге: он представляет собой спи-

сок всех слов, встречающихся в значениях столбцов, со ссылками на те значения, в которых

каждое слово содержится.

Вместо ключевого слова FULLTEXT можно использовать его синонимы – выражения

FULLTEXT INDEX или FULLTEXT KEY.

• ***SPATIAL*** [<Имя индекса>] (<Список столбцов>).

Создает индекс для поиска по пространственным и географическим значениям, кото-

рые остаются за рамками нашего рассмотрения.

• [***CONSTRAINT*** <Имя внешнего ключа>].

*FOREIGN KEY [<Имя индекса>] (<Список столбцов>)*

*REFERENCES <Имя родительской таблицы>*

*(<Список столбцов первичного ключа родительской таблицы>)*

*[<Правила поддержания целостности связи>]*

Определяет внешний ключ таблицы (внешние ключи мы рассматривали в главе 1).

Настроив внешний ключ, мы тем самым создадим связь между данной (дочерней) табли-

цей и родительской таблицей. Внешние ключи поддерживаются только для таблиц c типом

InnoDB (причем и дочерняя, и родительская таблица должны иметь тип InnoDB), для осталь-

ных типов таблиц выражение FOREIGN KEY игнорируется.

Столбцы, составляющие внешний ключ, должны иметь типы, аналогичные типам

столбцов первичного ключа в родительской таблице. Для числовых столбцов должен

совпадать размер и знак, для символьных – кодировка и правило сравнения значений.

Столбцы с типом TINYBLOB, TINYTEXT, BLOB, TEXT, MEDIUMBLOB, MEDIUMTEXT,

LONGBLOB и LONGTEXT не могут входить во внешний ключ.

Имя внешнего ключа и имя индекса указывать не обязательно. Если вы не зададите эти

имена, они будут автоматически сгенерированы.

Для операций удаления вы можете указать одно из следующих выражений:

• ***ON DELETE CASCADE*** – каскадное удаление строк дочерней таблицы (строка роди-

тельской таблицы удаляется вместе со всеми ссылающимися на нее строками дочерней

таблицы);

• ***ON DELETE SET NULL*** – обнуление значений внешнего ключа в соответствующих

строках дочерней таблицы;

• ***ON DELETE RESTRICT*** или ***ON DELETE NO ACTION*** (в MySQL эти выражения

являются синонимами) – запрет удаления строк родительской таблицы при наличии ссыла-

ющихся на них строк дочерней таблицы.

Если вы не задали правило поддержания целостности для операций удаления, по умол-

чанию используется правило ON DELETE RECTRICT.

Для операций изменения строк родительской таблицы вы можете указать одно из следующих выражений:

• ***ON UPDATE CASCADE*** – каскадное обновление значений внешнего ключа дочерней

таблицы (вместе со значением первичного ключа в строке родительской таблицы изменяется

значение внешнего ключа во всех ссылающихся на нее строках дочерней таблицы);

• ***ON UPDATE SET NULL*** – обнуление значений внешнего ключа в соответствующих

строках дочерней таблицы;

• ***ON UPDATE RECTRICT*** или ***ON UPDATE NO ACTION*** (в MySQL эти выражения

являются синонимами) – запрет изменения значений первичного ключа в строках родитель-

ской таблицы при наличии ссылающихся на них строк дочерней таблицы.

Если вы не задали правило поддержания целостности для операций изменения, по

умолчанию используется правило ON UPDATE RECTRICT.

**Опциональные свойства таблицы**

При создании таблицы указывать опциональные свойства не обязательно. Тем не

менее, рассмотрим некоторые свойства, которые вы можете задать для таблицы.

• ***ENGINE*** <Тип таблицы>.

Основными типами таблиц являются InnoDB и MyISAM.

Таблицы InnoDB обеспечивают поддержку транзакций (транзакции мы рассматривали

в главе 1, когда обсуждали понятие целостности данных) и блокировок отдельных строк,

благодаря которым обеспечивается высокая производительность операций изменения дан-

ных в многопользовательском режиме.

Кроме того, как мы увидели в предыдущем подразделе, в таблицах InnoDB можно

настроить внешние ключи для поддержания целостности связей между таблицами.

Таблицы MyISAM не поддерживают объединение нескольких операций в единую

транзакцию, поэтому, в частности, невозможно автоматическое поддержание целостности

связей между такими таблицами. Однако таблицы MyISAM также часто используются: их

преимуществом является высокая скорость выполнения поисковых запросов и меньшая

нагрузка на системные ресурсы.

Если требуется создать таблицу с типом MyISAM, добавим в команду создания

таблицы выражение ENGINE MyISAM. Если же при настройке вы предпочли вариант Non-

Transactional Database Only (Нетранзакционная база данных), то по умолчанию применяется

тип таблиц MyISAM, а таблицы типа InnoDB не поддерживаются.

• ***AUTO\_INCREMENT*** <Начальное значение>.

Задание этого свойства для таблицы, в которой есть столбец со свойством

AUTO\_INCREMENT, позволяет начать нумерацию в этом столбце не с единицы, а с указан-

ного вами значения. Например, если номера заказов должны начинаться с 1000, нужно в

команду создания таблицы Orders (см. листинг 2.4) включить параметр AUTO\_INCREMENT

1000.

• ***CHARACTER SET*** <Имя кодировки>.

Данный параметр определяет кодировку по умолчанию для символьных столбцов

таблицы.

Если не задана кодировка для таблицы, то по умолчанию используется кодировка, уста-

новленная для базы данных. Если и для базы данных кодировка не была указана, то исполь-

зуется кодировка, установленная по умолчанию при настройке MySQL.

• ***COLLATE*** <Имя правила сравнения>.

Данный параметр определяет правило сравнения значений, используемое по умолча-

нию для символьных столбцов таблицы.

• ***CHECKSUM*** 1.

Данный параметр включает проверку контрольной суммы для строк таблицы типа

MyISAM, что позволяет быстро обнаруживать поврежденные таблицы.

• ***COMMENT*** 'Текст комментария'.

Произвольное текстовое описание таблицы длиной до 60 символов. Например, описа-

ние для таблицы Customers (Клиенты) можно задать, включив в команду создания таблицы

параметр

***COMMENT*** *'Сведения о клиентах'*

Прочие опциональные параметры таблицы либо используются в целях оптимизации,

либо относятся к типам таблиц, которые в данной книге не рассматриваются.