

# **Chapter 11**

## **How to create databases, tables, and indexes**

# Objectives

## Applied

- Given the design for a database, write the DDL statements to create the tables, constraints, and indexes that are required.
- Write a script that includes all of the DDL statements for creating the tables of a database.
- Use MySQL Workbench to work with the columns, data, constraints, and indexes, for a table.

# Objectives (cont.)

## Knowledge

- Describe how each of these types of constraints restricts the values that can be stored in a table: not null, unique, primary key, and foreign key.
- Describe the difference between a column-level constraint and a table-level constraint.
- Describe the use of an index.
- Describe the use of a script for creating the tables of a database.
- Describe two character sets that are commonly used with MySQL and the pros and cons of each character set.
- Describe how a collation works with a character set.
- Describe two storage engines that are commonly used with MySQL and the pros and cons of each engine.

# How to use the CREATE DATABASE statement

## Syntax

```
CREATE DATABASE [IF NOT EXISTS] db_name
```

**Attempt to create a database named AP**

```
CREATE DATABASE ap
```

**Create a database named AP only if it doesn't exist**

```
CREATE DATABASE IF NOT EXISTS ap
```

# How to use the DROP DATABASE statement

## Syntax

```
DROP DATABASE [IF EXISTS] db_name
```

**Attempt to drop a database named AP**

```
DROP DATABASE ap
```

**Drop a database named AP only if it exists**

```
DROP DATABASE IF EXISTS ap
```

# How to use the USE statement

## Syntax

```
USE db_name
```

Select a database named AP

```
USE ap
```

# The syntax of the CREATE TABLE statement

```
CREATE TABLE [db_name.]table_name
(
    column_name_1 data_type [column_attributes]
    [, column_name_2 data_type [column_attributes]]...
    [, table_level_constraints]
)
```

## Common column attributes

- NOT NULL
- UNIQUE
- DEFAULT default\_value
- AUTO\_INCREMENT

## A statement that creates a table without column attributes

```
CREATE TABLE vendors
(
    vendor_id      INT,
    vendor_name    VARCHAR(50)
)
```

## A statement that creates a table with column attributes

```
CREATE TABLE vendors
(
    vendor_id      INT              NOT NULL      UNIQUE
                                AUTO_INCREMENT,
    vendor_name    VARCHAR(50)      NOT NULL      UNIQUE
)
```

## Another statement that creates a table with column attributes

```
CREATE TABLE invoices
(
    invoice_id      INT          NOT NULL UNIQUE,
    vendor_id       INT          NOT NULL,
    invoice_number  VARCHAR(50)  NOT NULL,
    invoice_date    DATE,
    invoice_total   DECIMAL(9,2) NOT NULL,
    payment_total   DECIMAL(9,2) DEFAULT 0
)
```

# The syntax of a column-level primary key constraint

```
column_name data_type PRIMARY KEY column_attributes
```

## A table with column-level constraints

```
CREATE TABLE vendors
(
    vendor_id      INT          PRIMARY KEY      AUTO_INCREMENT ,
    vendor_name    VARCHAR(50)   NOT NULL        UNIQUE
)
```

## The syntax of a table-level primary key constraint

```
[CONSTRAINT [constraint_name]]  
PRIMARY KEY (column_name_1[, column_name_2]...)
```

## A table with table-level constraints

```
CREATE TABLE vendors  
(  
    vendor_id      INT          AUTO_INCREMENT,  
    vendor_name    VARCHAR(50)   NOT NULL,  
    CONSTRAINT vendors_pk PRIMARY KEY (vendor_id),  
    CONSTRAINT vendor_name_uq UNIQUE (vendor_name)  
)
```

## A table with a two-column primary key constraint

```
CREATE TABLE invoice_line_items
(
    invoice_id                INT          NOT NULL,
    invoice_sequence           INT          NOT NULL,
    line_item_description      VARCHAR(100) NOT NULL,
    CONSTRAINT line_items_pk
        PRIMARY KEY (invoice_id, invoice_sequence)
)
```

## The syntax of a column-level foreign key constraint

```
[CONSTRAINT] REFERENCES table_name (column_name)  
[ON DELETE {CASCADE|SET NULL}]
```

## A table with a column-level foreign key constraint

```
CREATE TABLE invoices  
(  
    invoice_id      INT      PRIMARY KEY,  
    vendor_id       INT      REFERENCES vendors (vendor_id),  
    invoice_number  VARCHAR(50) NOT NULL      UNIQUE  
)
```

# The syntax of a table-level foreign key constraint

```
[CONSTRAINT constraint_name]
  FOREIGN KEY (column_name_1[, column_name_2]...)
    REFERENCES table_name (column_name_1
                           [, column_name_2]...)
  [ON DELETE {CASCADE|SET NULL}]
```

## A table with a table-level foreign key constraint

```
CREATE TABLE invoices
(
  invoice_id      INT          PRIMARY KEY,
  vendor_id       INT          NOT NULL,
  invoice_number  VARCHAR(50)   NOT NULL      UNIQUE,
  CONSTRAINT invoices_fk_vendors
    FOREIGN KEY (vendor_id)
      REFERENCES vendors (vendor_id)
)
```

## An INSERT statement that fails because a related row doesn't exist

```
INSERT INTO invoices  
VALUES (1, 1, '1')
```

## The response from the system

```
Error Code: 1452. Cannot add or update a child row: a  
foreign key constraint fails ('ex'.'invoices', CONSTRAINT  
'invoices_fk_vendors' FOREIGN KEY ('vendor_id')  
REFERENCES 'vendors' ('vendor_id'))
```

## A constraint that uses the ON DELETE clause

```
CONSTRAINT invoices_fk_vendors  
    FOREIGN KEY (vendor_id) REFERENCES vendors (vendor_id)  
    ON DELETE CASCADE
```

## Terms to know

- Constraint
- Column-level constraint
- Table-level constraint
- Not null constraint
- Unique constraint
- Primary key constraint
- Foreign key constraint

## The syntax for modifying the columns of a table

```
ALTER TABLE [db_name.]table_name
{
  ADD          column_name data_type [column_attributes] |
  DROP COLUMN column_name |
  MODIFY       column_name data_type [column_attributes]
}
```

## A statement that adds a new column

```
ALTER TABLE vendors  
ADD last_transaction_date DATE
```

## A statement that drops a column

```
ALTER TABLE vendors  
DROP COLUMN last_transaction_date
```

## A statement that changes the length of a column

```
ALTER TABLE vendors  
MODIFY vendor_name VARCHAR(100) NOT NULL UNIQUE
```

## A statement that changes the type of a column

```
ALTER TABLE vendors  
MODIFY vendor_name CHAR(100) NOT NULL UNIQUE
```

## A statement that changes the default value

```
ALTER TABLE vendors  
MODIFY vendor_name VARCHAR(100) NOT NULL  
                      DEFAULT 'New Vendor'
```

## A statement that fails because it would lose data

```
ALTER TABLE vendors  
MODIFY vendor_name VARCHAR(10) NOT NULL UNIQUE
```

### The response from the system

```
Error Code: 1265. Data truncated for column 'vendor_name'  
at row 1
```

# The syntax for modifying the constraints of a table

```
ALTER TABLE [dbname.]table_name
{
  ADD      PRIMARY KEY constraint_definition |
  ADD      [CONSTRAINT constraint_name]
  FOREIGN KEY constraint_definition |
  DROP      PRIMARY KEY |
  DROP      FOREIGN KEY constraint_name
}
```

## A statement that adds a primary key constraint

```
ALTER TABLE vendors  
ADD PRIMARY KEY (vendor_id)
```

## A statement that adds a foreign key constraint

```
ALTER TABLE invoices  
ADD CONSTRAINT invoices_fk_vendors  
FOREIGN KEY (vendor_id) REFERENCES vendors (vendor_id)
```

## A statement that drops a primary key constraint

```
ALTER TABLE vendors  
DROP PRIMARY KEY
```

## A statement that drops a foreign key constraint

```
ALTER TABLE invoices  
DROP FOREIGN KEY invoices_fk_vendors
```

## A statement that renames a table

```
RENAME TABLE vendors TO vendor
```

## A statement that deletes all data from a table

```
TRUNCATE TABLE vendor
```

## A statement that deletes a table from the current database

```
DROP TABLE vendor
```

## A statement that qualifies the table to be deleted

```
DROP TABLE ex.vendor
```

## A statement that returns an error due to a foreign key reference

```
DROP TABLE vendors
```

### The response from the system

```
Error Code: 1217. Cannot delete or update a parent row: a  
foreign key constraint fails
```

# The syntax of the CREATE INDEX statement

```
CREATE [UNIQUE] INDEX index_name  
    ON [dbname.]table_name (column_name_1 [ASC|DESC] [,  
                            column_name_2 [ASC|DESC]]...)
```

## A statement that creates an index based on a single column

```
CREATE INDEX invoices_invoice_date_ix  
    ON invoices (invoice_date)
```

## A statement that creates an index based on two columns

```
CREATE INDEX invoices_vendor_id_invoice_number_ix  
    ON invoices (vendor_id, invoice_number)
```

## A statement that creates a unique index

```
CREATE UNIQUE INDEX vendors_vendor_phone_ix  
ON vendors (vendor_phone)
```

## A statement that creates an index that's sorted in descending order

```
CREATE INDEX invoices_invoice_total_ix  
ON invoices (invoice_total DESC)
```

## A statement that drops an index

```
DROP INDEX vendors_vendor_phone_ix ON vendors
```

# The SQL script that creates the AP database

```
-- create the database
DROP DATABASE IF EXISTS ap;
CREATE DATABASE ap;

-- select the database
USE ap;

-- create the tables
CREATE TABLE general_ledger_accounts
(
    account_number          INT           PRIMARY KEY,
    account_description     VARCHAR(50)   UNIQUE
);

CREATE TABLE terms
(
    terms_id                INT           PRIMARY KEY,
    terms_description        VARCHAR(50)   NOT NULL,
    terms_due_days           INT           NOT NULL
);
```

## The SQL script (continued)

```
CREATE TABLE vendors
(
    vendor_id          INT      PRIMARY KEY AUTO_INCREMENT,
    vendor_name        VARCHAR(50)      NOT NULL UNIQUE,
    vendor_address1   VARCHAR(50),
    vendor_address2   VARCHAR(50),
    vendor_city        VARCHAR(50)      NOT NULL,
    vendor_state       CHAR(2)          NOT NULL,
    vendor_zip_code   VARCHAR(20)      NOT NULL,
    vendor_phone       VARCHAR(50),
    vendor_contact_last_name  VARCHAR(50),
    vendor_contact_first_name  VARCHAR(50),
    default_terms_id   INT      NOT NULL,
    default_account_number  INT      NOT NULL,
    CONSTRAINT vendors_fk_terms
        FOREIGN KEY (default_terms_id)
        REFERENCES terms (terms_id),
    CONSTRAINT vendors_fk_accounts
        FOREIGN KEY (default_account_number)
        REFERENCES general_ledger_accounts (account_number)
);
```

## The SQL script (continued)

```
CREATE TABLE invoices
(
    invoice_id      INT      PRIMARY KEY      AUTO_INCREMENT,
    vendor_id       INT      NOT NULL,
    invoice_number  VARCHAR(50)      NOT NULL,
    invoice_date    DATE      NOT NULL,
    invoice_total   DECIMAL(9,2)      NOT NULL,
    payment_total   DECIMAL(9,2)      NOT NULL      DEFAULT 0,
    credit_total    DECIMAL(9,2)      NOT NULL      DEFAULT 0,
    terms_id        INT      NOT NULL,
    invoice_due_date DATE      NOT NULL,
    payment_date    DATE,
    CONSTRAINT invoices_fk_vendors
        FOREIGN KEY (vendor_id)
        REFERENCES vendors (vendor_id),
    CONSTRAINT invoices_fk_terms
        FOREIGN KEY (terms_id)
        REFERENCES terms (terms_id)
) ;
```

## The SQL script (continued)

```
CREATE TABLE invoice_line_items
(
    invoice_id                INT          NOT NULL,
    invoice_sequence           INT          NOT NULL,
    account_number              INT          NOT NULL,
    line_item_amount            DECIMAL(9,2) NOT NULL,
    line_item_description      VARCHAR(100) NOT NULL,
    CONSTRAINT line_items_pk
        PRIMARY KEY (invoice_id, invoice_sequence),
    CONSTRAINT line_items_fk_invoices
        FOREIGN KEY (invoice_id)
        REFERENCES invoices (invoice_id),
    CONSTRAINT line_items_fk_accounts
        FOREIGN KEY (account_number)
        REFERENCES general_ledger_accounts (account_number)
);

-- create an index
CREATE INDEX invoices_invoice_date_ix
    ON invoices (invoice_date DESC);
```

# The column definitions for the Invoices table

The screenshot shows the MySQL Workbench interface with the 'invoices' table selected in the Navigator. The main pane displays the table structure with the following columns:

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
invoice_id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
vendor_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
invoice_number	VARCHAR(50)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
invoice_date	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
invoice_total	DECIMAL(9,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
payment_total	DECIMAL(9,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'0.00'				
credit_total	DECIMAL(9,2)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'0.00'				
terms_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
invoice_due_date	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
payment_date	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Below the table structure, a detailed view of the first column ('invoice\_id') is shown:

Column Name:	invoice_id	Data Type:	INT(11)
Collation:	Table Default	Default:	
Comments:			
<input checked="" type="checkbox"/> Primary <input checked="" type="checkbox"/> Not Null <input type="checkbox"/> Unique <input type="checkbox"/> Binary <input type="checkbox"/> Unsigned <input type="checkbox"/> Zero Fill <input checked="" type="checkbox"/> Auto Increment			

Buttons at the bottom include 'Apply' and 'Revert'.

# The indexes for the Invoices table

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the schema structure under the "ap" database, including tables like general\_ledger\_accounts, invoice\_archive, invoice\_line\_items, invoices, terms, vendor\_contacts, and vendors.
- Query 1 - invoices - Table:** This tab is active, displaying information about the "invoices" table.
  - Table Name:** invoices
  - Schema:** ap
  - Collation:** utf8 - default collation
  - Engine:** InnoDB
  - Comments:** (empty)
- Indexes:** A table showing the current indexes for the "invoices" table.

Index Name	Type
PRIMARY	PRIMARY
invoices_fk_vendors	INDEX
invoices_fk_terms	INDEX
invoices_invoice_date...	INDEX
- Index Columns:** A table showing the columns and their order for each index.

Column	#	Order	Le
invoice_id			ASC
vendor_id			ASC
invoice_number			ASC
<input checked="" type="checkbox"/> invoice_date	1		ASC
invoice_total			ASC
payment_total			ASC
credit_total			ASC
terms_id			ASC
invoice_due_date			ASC
payment_date			ASC
- Index Options:** Fields for configuring storage type, key block size, and parser.
- Index Comment:** A text area for comments.
- Buttons:** Apply and Revert.

# The foreign keys for the Invoices table

The screenshot shows the MySQL Workbench interface with the 'invoicing' schema selected. The central pane displays the 'invoicing - Table' configuration for the 'invoices' table. The table name is set to 'invoices', the schema is 'ap', the collation is 'utf8 - default collation', and the engine is 'InnoDB'. The 'Foreign Keys' tab is selected, showing two foreign key definitions:

Foreign Key Name	Referenced Table	Column	Referenced Column
invoices_fk_terms	'ap`.`terms'	invoice_id	terms_id
invoices_fk_vendors	'ap`.`vendors'	vendor_id	vendor_id

The 'On Update' and 'On Delete' options are both set to 'RESTRICT'. There is also a checkbox for 'Skip in SQL generation' which is unchecked. The 'Foreign Key Comment' field is empty.

The left sidebar shows the 'Navigator' with the 'Tables' section expanded, listing 'general\_ledger\_accounts', 'invoice\_archive', 'invoice\_line\_items', 'invoices', 'terms', 'vendor\_contacts', and 'vendors'. Below this are 'Views', 'Stored Procedures', and 'Functions'. The bottom left pane shows the table structure for 'invoices' with columns: invoice\_id (int(11) AI PK), vendor\_id (int(11)), invoice\_number (varchar(50)), invoice\_date (date), invoice\_total (decimal(9,2)), payment\_total (decimal(9,2)), credit\_total (decimal(9,2)), terms\_id (int(11)), invoice\_due\_date (date), and payment\_date (date).

## Two commonly used character sets

- latin1
- utf8

## Four collations for the latin1 character set

- latin1\_swedish\_ci
- latin1\_general\_ci
- latin1\_general\_cs
- latin1\_bin

## Four collations for the utf8 character set

- utf8\_general\_ci
- utf8\_unicode\_ci
- utf8\_spanish\_ci
- utf8\_bin

## Collation names

- If the name ends with ci, the collation is case-insensitive.
- If the name ends with cs, the collation is case-sensitive.
- If the name ends with bin, the collation is binary.

# How to view all available character sets for a server

`SHOW CHARSET`

	Charset	Description	Default collation	Maxlen
	latin1	cp1252 West European	latin1_swedish_ci	1
	latin2	ISO 8859-2 Central European	latin2_general_ci	1
	swe7	7bit Swedish	swe7_swedish_ci	1
	ascii	US ASCII	ascii_general_ci	1
	ujis	EUC-JP Japanese	ujis_japanese_ci	3

# How to view a specific character set

`SHOW CHARSET LIKE 'latin1'`

# How to view all available collations for a server

`SHOW COLLATION`

	Collation	Charset	Id	Default	Compiled	Sortlen
	latin1_swedish_ci	latin1	8	Yes	Yes	1
	latin1_danish_ci	latin1	15		Yes	1
	latin1_german2_ci	latin1	31		Yes	2
	latin1_bin	latin1	47		Yes	1
	latin1_general_ci	latin1	48		Yes	1
	latin1_general_cs	latin1	49		Yes	1
	latin1_spanish_ci	latin1	94		Yes	1
	latin2_czech_cs	latin2	2		Yes	4
	latin2_general_ci	latin2	9	Yes	Yes	1

# How to view all available collations for a specific character set

`SHOW COLLATION LIKE 'latin1%'`

## **How to view the default character set for a server**

```
SHOW VARIABLES LIKE 'character_set_server'
```

## **How to view the default collation for a server**

```
SHOW VARIABLES LIKE 'collation_server'
```

## **How to view the default character set for a database**

```
SHOW VARIABLES LIKE 'character_set_database'
```

## **How to view the default collation for a database**

```
SHOW VARIABLES LIKE 'collation_database'
```

## How to view the character set and collation for all the tables in a database

```
SELECT table_name, table_collation  
FROM information_schema.tables  
WHERE table_schema = 'ap'
```

	table_name	table_collation
	invoice_line_items	utf8_general_ci
	invoices	utf8_general_ci
	terms	utf8_general_ci

# The clauses used to specify a character set and collation

```
[CHARSET character_set] [COLLATE collation]
```

## How to specify a character set and collation at the database level

### For a new database

```
CREATE DATABASE ar CHARSET latin1  
                      COLLATE latin1_swedish_ci
```

### For an existing database

```
ALTER DATABASE ar CHARSET utf8 COLLATE utf8_general_ci
```

### For an existing database using the CHARSET clause only

```
ALTER DATABASE ar CHARSET utf8
```

### For an existing database using the COLLATE clause only

```
ALTER DATABASE ar COLLATE utf8_general_ci
```

# How to specify a character set and collation at the table level

## For a new table

```
CREATE TABLE employees
(
    emp_id          INT           PRIMARY KEY,
    emp_name        VARCHAR(25)
)
CHARSET latin1 COLLATE latin1_swedish_ci
```

## For an existing table

```
ALTER TABLE employees
CHARSET utf8 COLLATE utf8_general_ci
```

# How to specify a character set and collation at the column level

## For a column in a new table

```
CREATE TABLE employees
(
    emp_id      INT          PRIMARY KEY,
    emp_name    VARCHAR(25)  CHARSET latin1
                           COLLATE latin1_swedish_ci
)
```

## For a column in an existing table

```
ALTER TABLE employees
MODIFY emp_name VARCHAR(25) CHARSET utf8
                           COLLATE utf8_general_ci
```

## Two commonly used storage engines

- InnoDB
- MyISAM

# How to view all storage engines for a server

**SHOW ENGINES**

Engine	Support	Comment	Transactions	XA	Savepoints
FEDERATED	NO	Federated MySQL storage engine	NULL	NULL	NULL
MRG_MYISAM	YES	Collection of identical MyISAM tables	NO	NO	NO
MyISAM	YES	MyISAM storage engine	NO	NO	NO
BLACKHOLE	YES	/dev/null storage engine (anything you write to ...)	NO	NO	NO
CSV	YES	CSV storage engine	NO	NO	NO
MEMORY	YES	Hash based, stored in memory, useful for temp...	NO	NO	NO
ARCHIVE	YES	Archive storage engine	NO	NO	NO
InnoDB	DEFAULT	Supports transactions, row-level locking, and fo...	YES	YES	YES
PERFORMANCE_SCHEMA	YES	Performance Schema	NO	NO	NO

# How to view the default storage engine for a server

**SHOW VARIABLES LIKE 'storage\_engine'**

# How to view the storage engine for all the tables in a database

```
SELECT table_name, engine  
FROM information_schema.tables  
WHERE table_schema = 'ap'
```

	table_name	engine
	invoice_line_items	InnoDB
	invoices	InnoDB
	terms	InnoDB

## The clause used to specify a storage engine

`ENGINE = engine_name`

### How to specify a storage engine for a table

#### For a new table

```
CREATE TABLE product_descriptions
(
    product_id          INT           PRIMARY KEY,
    product_description VARCHAR(200)
)
ENGINE = MyISAM
```

#### For an existing table

```
ALTER TABLE product_descriptions ENGINE = InnoDB
```

### How to set the default storage engine for the current session

```
SET SESSION storage_engine = InnoDB
```