TOOL

SQL Commands for Working with Databases

SQL can be very powerful when working with relational databases. Use this tool as a reference for commands commonly used when working with databases.

Command	Description and Example(s)
`CREATE DATABASE <database_name>;`	Create a database.
`RENAME DATABASE <old_database_name> TO <new_database_name>;`	Rename a database.
`DROP DATABASE database_name;`	Permanently delete a database.

`CREATE TABLE <table_name> (`
`  <column1> <datatype>,`
`  <column2> <datatype>,`
`  …`
`);`
	Create a table.

`ALTER TABLE <table_name> <action>;`	Alter a table.
	
	For example, you could add a column with the command:
	`ALTER TABLE <table_name>`
	`ADD COLUMN <column_name> <datatype>;`
	
	Or delete a column with the command:
	`ALTER TABLE <table_name>`
	`DROP COLUMN <column_name>;`

© 2023 Cornell University

Command	Description and Example(s)

`CONSTRAINT <pk_name> PRIMARY KEY (<pk_column_list>);`	Add this command to your SQL code when creating or altering tables to name your Primary Key.
	
	For example, you may add it when creating a table:
	`CREATE TABLE <table_name> (`
	`  <column_1> <datatype>,`
	`  <column_2> <datatype>,`
	`  CONSTRAINT <pk_name> PRIMARY KEY`
	`  (<pk_column_list>);`
	`);`

`CONSTRAINT <fk_name> FOREIGN KEY (<column_name>) REFERENCES <key_table>(<key_column>);`	Add this command to your SQL code when creating or altering tables to name your Foreign Key.
	
	For example, you may add it when creating a table:
	`CREATE TABLE <table_name> (`
	`  <column_1> <datatype>,`
	`  <column_2> <datatype>,`
	`  CONSTRAINT <fk_name> FOREIGN KEY`
	`  (<column_name>) REFERENCES`
	`  <key_table>`
	`  (<key_column>);`
	`);`
	
	When using this command with the `ALTER TABLE` command, you should put the word `ADD` before it:
	`ALTER TABLE <table_name> ADD CONSTRAINT <fk_name> FOREIGN KEY (<column_name>) REFERENCES <key_table>(<key_column>);`

`CREATE INDEX <index_name> ON <table_name> (<column1>, <column2>, …);`	Create an index.

© 2023 Cornell University

Command	Description and Example(s)
`INSERT INTO <table_name> (<column1>, <column2>, …)` `VALUES (<value1>, <value2>, …);`	Add new data rows into a table; note that the values must be in the same order as the columns.

`UPDATE <table_name> SET (<column1>=<value1>,`
`<column2>=<value2>, …)`
`WHERE <conditions>;`
	Modify existing values in rows.

`DELETE FROM <table_name>` `WHERE <conditions>;`	Remove rows from a table. The condition is very important in this command; if you provide no condition, all rows in the table will be deleted!
`COUNT (<condition>)`	Return the number of fields matching a given condition.
`COUNT (DISTINCT <column_name>);`	Return the number of fields matching a given condition, ignoring duplicates.

`AND`
`OR`
`NOT`
`<>`
`IS NULL`
`IS NOT NULL`
	Use these operations to improve the logic of your queries and check for null or not-null values.
	
	The `AND` operator displays a record if all the conditions separated by `AND` are TRUE.
	
	The `OR` operator displays a record if any of the conditions separated by `OR` are TRUE.
	
	The `NOT` operator displays a record if any of the conditions are NOT TRUE.
	
	Use `<>` as a "does not equal" sign.

`GROUP BY <column_name>, …` `HAVING <condition>`	In conjunction with `SELECT…FROM`, `GROUP BY` returns your selection categorized into groups based on the values in a specified column. Use `HAVING` in place of `WHERE` to specify a condition when working within aggregate functions.

© 2023 Cornell University

Command	Description and Example(s)
`SELECT <column1>, <column2>, ...` `FROM <table_name>;`	Select data within a database.
`ORDER BY <column_name>, … [ASC|DESC];`	In conjunction with `SELECT…FROM`, `ORDER BY` returns your selection ordered according to the values in a specified column. Use `ASC` or `DESC` to specify ascending or descending order.

`[INNER|LEFT|RIGHT|FULL] JOIN <table1>`
`ON <table2.column_name> =`
`<table1.column_name>`
`WHERE <condition>;`
	Combine rows from two tables.
	
	Use `INNER`, `LEFT`, `RIGHT`, or `FULL` to specify the type of overlap between tables.

`SELECT <column_name> AS <alias>`
`FROM <table_name>;`
 	
`SELECT <table_name> AS <alias>`
`FROM <database_name>;`
	Rename a column or table with an alias that can be referenced only during that query.

`USE <database_name>;`	Connect the user to a database.

`SELECT <column_name>, … FROM table1`
`UNION`
`SELECT <column_name>, … FROM table2;`
	Create a combined set from the results of two or more `SELECT` commands.

MySQL Specific Commands

It is helpful to view the tables you are working with; use the table below as a reference when viewing tables with MySQL.

Command	Description and Example(s)

`SHOW COLUMNS FROM <table_name>;`	Show a list of all existing columns in a table.
	
	To show only columns that meet a certain condition, use the `WHERE` clause with this command:
	`SHOW COLUMNS FROM <table_name>`
	`WHERE <condition>;`

© 2023 Cornell University

Command	Description and Example(s)
`SHOW TABLES FROM <database_name>;`	Show a list of all existing tables in a database. To show only tables that meet a certain condition, use the `WHERE` clause with this command.
`SHOW FULL`	Use this command in conjunction with either of the two previous commands to return more information about your columns or tables. **Note:** For the purposes of this course, this is not a typical command. A command like this would be more useful for a database administrator.
`SHOW INDEXES FROM <table_name>;`	Show a list of all existing tables in a database. To show only tables that meet a certain condition, use the `WHERE` clause with this command. **Note:** The `SHOW FULL` command does not work here.

`RENAME TABLE <old_table_name>`
`TO <new_table_name>;`
	Change a table's name using the `RENAME` command.
	
	Note: You can also use the `ALTER` command to rename a table:
	`ALTER TABLE <old_table_name>`
	`RENAME <new_table_name>;`

`SELECT * FROM <table_name>` `LIMIT X;`	Retrieve the top X rows of a database. To sort the order of results, use the `ORDER BY` clause with this command.

© 2023 Cornell University