Index

aggregate functions

Using the HAVING Clause

1. Filtering **aggregate functions** With The HAVING Clause

Aggregate Functions

1. **aggregate functions**
2. Table 1. **aggregate functions** List

and

The DELETE Clause

1. ... WHERE name = 'san felipe' **and** countrycode = 'chl';
2. WHERE name = 'san felipe' **and** countrycode = 'chl';

Grouping Data

1. Filtering With WHERE **and** HAVING

Using the HAVING and WHERE Clauses Together

1. … that includes both the HAVING **and** WHERE clause in the same SQL statement.

COUNT(column_name) and COUNT(*)

1. COUNT(column_name) **and** COUNT(*)

Introduction

1. … databases that you can download **and** install in your local MySQL instance.…
2. … will include SQL design basics **and** guidance on how to install MySQL **and** MySQL…

How to Retrieve Data From a Single Table

1. LIKE **and** REGEXP Operators
2. **and**, OR, NOT Logical Operators
3. **and**
4. Separates two string patterns **and** matches either one
5. … countryWHERE region = 'caribbean'**and** population > 100000ORDER BY population…
6. … countryWHERE name BETWEEN "Aruba" **and** "Bahamas";
7. Table 4. Operators **and** precendence order
8. (a **and** b) –If both a **and** b are present, item is included

LIKE and REGEXP Operators

1. LIKE **and** REGEXP Operators
2. Separates two string patterns **and** matches either one

Arithmetic Operators

1. Table 4. Operators **and** precedence order

IS NULL, BETWEEN, IN Operators

1. … countryWHERE name BETWEEN "Aruba" **and** "Bahamas";

AND, OR, NOT Logical Operators

1. **and**, OR, NOT Logical Operators
2. **and**
3. … countryWHERE region = 'caribbean'**and** population > 100000ORDER BY population…
4. (a **and** b) –If both a **and** b are present, item is included

The JOIN Clause

1. … table aliases of co for country **and** ci for city are defined in the FROM clause…

Date Functions

1. * Returns current local date **and** time.

String Functions

1. LOCATE(), **and** LENGTH() accept a string but return an integer. • SUBSTRING()…

arithmetic operators

How to Retrieve Data From a Single Table

1. **arithmetic operators**

Arithmetic Operators

1. **arithmetic operators**

avg

Simple GROUP BY Query

1. USE bike;SELECT category_id, **avg**(list_price)FROM productGROUP BY category_id
2. SELECT category_id, **avg**(list_price):

Improving the GROUP BY Query

1. … category_name, CONCAT('$', ROUND(**avg**(list_price),2)) AS 'Average List…
2. CONCAT('$', ROUND(**avg**(list_price),2)) AS 'Average List Price'

Using the HAVING Clause

1. HAVING **avg**(list_price) > 1000
2. USE bike;SELECT category_id, **avg**(list_price)FROM productGROUP BY category_idHAVING…

Using the HAVING and WHERE Clauses Together

1. USE bike;SELECT category_id, **avg**(list_price)FROM productWHERE model_year =…
2. HAVING **avg**(list_price) > 1000

Aggregate Functions

1. USE bike;SELECT **avg**(list_price), SUM(list_price), MIN(list_price), …
2. **avg**([DISTINCT] column_values)

between

How to Retrieve Data From a Single Table

1. **between** Operators
2. … IndepYearFROM countryWHERE name **between** "Aruba" and "Bahamas";

IS NULL, BETWEEN, IN Operators

1. **between** Operators
2. … IndepYearFROM countryWHERE name **between** "Aruba" and "Bahamas";

ceiling

Numeric Functions

1. FLOOR, **ceiling**, TRUNCATE
2. … list_price, FLOOR(list_price), **ceiling**(list_price), TRUNCATE(list_price,…
3. **ceiling**(number)
4. **ceiling**(6.2)
5. Table 6. FLOOR, **ceiling**, TRUNCATE functions

column

How to Retrieve Data From a Single Table

1. **column** Aliases
2. Show all **column**s
3. Comma separated list of **column** names
4. … previous example, we created a new **column** that was a calculated value. The problem…
5. **column** Name
6. Table 1. **column** Specifications
7. … then in quotes we put the new **column** alias of “People per square mile.”…

The Five Clauses of the SELECT Statement

1. Show all **column**s
2. Comma-separated list of **column** names
3. **column** Name
4. Table 1. **column** Specifications

Column Specifications

1. **column** Specifications
2. Show all **column**s
3. **column** Name
4. Comma separated list of **column** names
5. **column** Specifications

Column Aliases

1. **column** Aliases
2. … previous example, we created a new **column** that was a calculated value. The problem…
3. … then in quotes we put the new **column** alias of “People per square mile.”…

The JOIN Clause

1. … whole table name to qualify a **column**, you can use a table alias.

The INSERT Clause With a Column List

1. The INSERT Clause With a **column** List
2. … of an INSERT statement with a **column** list:

The INSERT Clause Without a Column List

1. The INSERT Clause Without a **column** List

Grouping Data

1. … rows of a result set based on **column**s or expressions separated by commas.

COUNT(column_name) and COUNT(*)

1. COUNT(**column**_name) and COUNT(*)

Aggregate Functions

1. SUM([DISTINCT] **column**_values)
2. MIN([DISTINCT] **column**_values)
3. The average of the non-null **column**s in the expression
4. MAX([DISTINCT] **column**_values)
5. COUNT([DISTINCT] **column**_values)
6. … highest value of the non-null **column**s in the expression
7. AVG([DISTINCT] **column**_values)
8. The total of the non-null **column**s in the expression
9. … lowest value off the non-null **column**s in the expression
10. The number of the non-null **column**s in the expression

column aliases

How to Retrieve Data From a Single Table

1. **column aliases**

Column Aliases

1. **column aliases**

column specifications

How to Retrieve Data From a Single Table

1. Table 1. **column specifications**

The Five Clauses of the SELECT Statement

1. Table 1. **column specifications**

Column Specifications

1. **column specifications**
2. **column specifications**

comparison operators

How to Retrieve Data From a Single Table

1. **comparison operators**
2. Table 5. **comparison operators**

Comparison Operators

1. **comparison operators**
2. Table 5. **comparison operators**

concat

String Functions

1. **concat**
2. USE world;SELECT **concat**(name, ', ', continent)FROM country;

Improving the GROUP BY Query

1. … bike;SELECT category_name, **concat**('$', ROUND(AVG(list_price),2)) AS 'Average…
2. **concat**('$', ROUND(AVG(list_price),2)) AS 'Average List Price'

count

The INSERT Clause With a Column List

1. (name, **count**ryCode, district, population)
2. … INTO city 3 (name, **count**ryCode, district, population) 4 VALUES 5…

The DELETE Clause

1. … WHERE name = 'san felipe' AND **count**rycode = 'chl';
2. WHERE name = 'san felipe' AND **count**rycode = 'chl';

COUNT(column_name) and COUNT(*)

1. **count**(column_name) and **count**(*)
2. USE bike;SELECT **count**(phone), **count**(*) FROM CUSTOMER

Using the DISTINCT Statement

1. ExampleUSE bike;SELECT **count**(list_price), **count**(DISTINCT list_price) FROM product;

The Subquery in an UPDATE statement

1. (SELECT **count**ryCode FROM **count**rylanguage WHERE population = 0)
2. 1 UPDATE **count**ry 2 SET GNPOld = 0.00 3 WHERE Code IN 4 …
3. UPDATE **count**ry

The Subquery In a Delete Statement

1. (SELECT code FROM **count**ry
2. … world;DELETE FROM city_bakWHERE **count**ryCode IN (SELECT code FROM **count**ry …
3. WHERE **count**ryCode IN

Benefits of Using Views

1. USE WORLD;CREATE VIEW city_**countr**y ASSELECT ci.name AS city_name, co.name AS…
2. CREATE VIEW city_**countr**y AS
3. … ci.name AS city_name, co.name AS **countr**y_name
4. JOIN **countr**y co
5. ON ci.**countr**yCode = co.Code;
6. Results by selecting from the city_**countr**y view:

Aggregate Functions

1. **count**(*)
2. … MIN(list_price), MAX(list_price), **count**(list_price), **count**(*)FROM product;
3. **count**([DISTINCT] column_values)

The Subquery In a SELECT Statement

1. WHERE **countr**yCode IN
2. … FROM city 4 WHERE **countr**yCode IN 5 (SELECT code 6 …
3. FROM **countr**y

How to Retrieve Data From a Single Table

1. USE world;SELECT nameFROM **countr**yWHERE name IN ('Aruba', 'Barbados', 'Cuba',…
2. … “People per square mile”FROM **countr**y;
3. SELECT name, IndepYearFROM **countr**yWHERE IndepYear IS NULL;
4. USE world;SELECT nameFROM **countr**yWHERE name REGEXP 'g[o,u]';
5. … world;SELECT name, populationFROM **countr**yWHERE population > 1000000;
6. … world;SELECT name, populationFROM **countr**yWHERE region = 'caribbean'AND population…
7. … "People per square mile"FROM **countr**y;
8. … world;SELECT name, IndepYearFROM **countr**yWHERE name BETWEEN "Aruba" and "Bahamas";
9. … DISTINCT continent, nameFROM **countr**yORDER BY continent;
10. USE world;SELECT nameFROM **countr**yWHERE name LIKE ‘A%’

The Five Clauses of the SELECT Statement

1. … name3 FROM city4 WHERE **countr**yCode = “AFG”5 ORDER BY name6…

LIKE and REGEXP Operators

1. USE world;SELECT nameFROM **countr**yWHERE name REGEXP 'g[o,u]';
2. USE world;SELECT nameFROM **countr**yWHERE name LIKE ‘A%’

Arithmetic Operators

1. … "People per square mile"FROM **countr**y;

Column Aliases

1. … “People per square mile”FROM **countr**y;

Comparison Operators

1. … world;SELECT name, populationFROM **countr**yWHERE population > 1000000;

IS NULL, BETWEEN, IN Operators

1. USE world;SELECT nameFROM **countr**yWHERE name IN ('Aruba', 'Barbados', 'Cuba',...
2. SELECT name, IndepYearFROM **countr**yWHERE IndepYear IS NULL;
3. ... world;SELECT name, IndepYearFROM **countr**yWHERE name BETWEEN "Aruba" and "Bahamas";

AND, OR, NOT Logical Operators

1. ... world;SELECT name, populationFROM **countr**yWHERE region = 'caribbean'AND population...

DISTINCT Clause

1. ... DISTINCT continent, nameFROM **countr**yORDER BY continent;

The JOIN Clause

1. ... “City Name”, co.name AS “**countr**y Name”
2. JOIN **countr**y co
3. ON ci.**countr**yCode = co.Code;
4. ... AS "City Name", 3 **countr**y.name AS "**countr**y Name" 4 FROM **countr**y 6...
5. ... aliases. The table aliases of co for **countr**y and ci for city are defined in...
6. ... Name", 3 co.name AS "**countr**y Name" 4 FROM city ci 5 ...

Joining More Than Two Tables

1. ON cl.**countr**yCode = ci.**countr**yCode;
2. JOIN **countr**ylanguage cl.
3. ... Name",3 co.name AS "**countr**y Name", 4 cl.language AS...

The OUTER JOIN Clause

1. ON c.code = cl.**countr**yCode
2. FROM **countr**y c LEFT JOIN **countr**ylanguage cl
3. ... c.continent, cl.language3 FROM **countr**y c LEFT JOIN **countr**ylanguage cl4 ON c.code...

How to Code a UNION

1. SELECT name, populationFROM **countr**yWHERE continent = 'Oceania'
2. ... name, populationFROM cityWHERE **countr**yCode = 'AUS'
3. ... name, population3 FROM city WHERE **countr**yCode = 'AUS'4 UNION5 SELECT name,...

Date Functions

1. ... DATE_FORMAT('2020-01-28', '%m/%d/%y')FROM **countr**y;

Numeric Functions

1. ... ROUND(LifeExpectancy) FROM world.**countr**y;

String Functions

1. ... CONCAT(name, ', ', continent)FROM **countr**y;

current_date

Date Functions

1. **current_date**()
2. **current_date**
3. … DATE('2020-01-01') AS 'DATE(), date only', **current_date** AS '**current_date**', CURRENT_TIME…

current_time

Date Functions

1. **current_time**
2. … CURRENT_DATE AS 'CURRENT_DATE', **current_time** AS '**current_time**', UTC_DATE…
3. **current_time**()

date

Date Functions

1. **date**_FORMAT
2. **date**_ADD
3. **date**DIFF
4. Current **date**/Time Functions
5. **date**, **date**TIME
6. Table 1. Current **date** Functions
7. CURRENT_**date**()
8. **date**
9. **date**(**date**)
10. CURRENT_**date**
11. Table 3. **date**_FORMAT Function
12. * Returns current local **date**
13. **date**_FORMAT
14. **date**/time
15. Table 2. **date**_ADD Function
16. **date**
17. SELECT NOW() AS 'NOW()', **date**('2020-01-01') AS '**date**(), **date** only', ...
18. • **date**s must be enclosed in quotes • You can pass a **date** or **date**TIME datatype...
19. SELECT **date**DIFF('2018-01-01', '2019-01-01') AS '**date** Difference';
20. * extracts the **date** from input. If time is included, the time is dropped.
21. **date**_FORMAT('2020-09-03', '%m/%d/%y')
22. ... world;SELECT name, continent, **date**_FORMAT('2020-01-28', '%m/%d/%y')FROM country;
23. * Returns current local **date** and time.
24. **date**/time
25. USE bike;SELECT order_**date**, **date**_ADD(order_**date**, INTERVAL 1 DAY) AS 'ORDER...
26. * Returns current UTC **date**.
27. **date**
28. **date**('2020-01-01 11:31:31')
29. **date**_ADD(**date**, interval expression unit)
30. * Returns current UTC **date**.
31. UTC_**date**()
32. UTC_**date**
33. **date**_ADD('2020-01-01', INTERVAL 1 DAY)
34. • Returns a **date** with a **date** or **date**TIME value equal to the original value...

The UPDATE Clause With a Column List

1. The UP**date** Clause
2. UP**date** city
3. 1 USE world; 2 UP**date** city 3 SET Population = 65000, district...

The Subquery in an UPDATE statement

1. The Subquery in an UP**date** statement
2. 1 UP**date** country 2 SET GNPOld = 0.00 3 WHERE Code IN 4 ...
3. UP**date** country

The Subquery In a Delete Statement

1. … Before you can run a DELETE or UP**date** statement without a WHERE clause, you…

Views That Allow UPDATE Statements

1. … Views That Can Be Used With an UP**date** Statement

Aggregate Functions

1. numeric, **date**, string
2. numeric, **date**, string

delete

The DELETE Clause

1. The **delete** Clause
2. 1 USE world;2 **delete** 3 FROM city 4 WHERE name = 'san felipe'…
3. **delete**

The Subquery In a Delete Statement

1. The Subquery in a **delete** statement
2. **delete** FROM city_bak
3. USE world;**delete** FROM city_bakWHERE CountryCode IN (SELECT code FROM country …
4. NOTE: Before you can run a **delete** or UPDATE statement without a WHERE clause,…

distinct

Using the DISTINCT Statement

1. Removing Duplicate Values With **distinct**
2. … bike;SELECT COUNT(list_price), COUNT(**distinct** list_price) FROM product;

Aggregate Functions

1. SUM([**distinct**] column_values)
2. MIN([**distinct**] column_values)
3. MAX([**distinct**] column_values)
4. COUNT([**distinct**] column_values)
5. AVG([**distinct**] column_values)

How to Retrieve Data From a Single Table

1. **distinct** Keyword
2. **distinct**
3. Table 7. **distinct** Keyword
4. SELECT **distinct** continent, nameFROM countryORDER BY continent;

DISTINCT Clause

1. **distinct** Keyword
2. **distinct**
3. Table 7. **distinct** Keyword
4. SELECT **distinct** continent, nameFROM countryORDER BY continent;

floor

Numeric Functions

1. **floor**, CEILING, TRUNCATE
2. **floor**(7.7)
3. USE bike;SELECT list_price, **floor**(list_price), CEILING(list_price), TRUNCATE(list_price,...
4. **floor**(number)
5. Table 6. **floor**, CEILING, TRUNCATE functions

group by

Grouping Data

1. Using the **group by** Clause
2. Table 1. **group by** Function
3. **group by**

Simple GROUP BY Query

1. **group by** category_id:
2. ... category_id, AVG(list_price)FROM product**group by** category_id

Improving the GROUP BY Query

1. Improving the **group by** Query
2. ... p.category_id = c.category_id**group by** category_nameORDER BY category_name;
3. **group by** category_name

Using the HAVING Clause

1. ... category_id, AVG(list_price)FROM product**group by** category_idHAVING AVG(list_price)...

Using the HAVING and WHERE Clauses Together

1. ... productWHERE model_year = 2016**group by** category_idHAVING AVG(list_price) > 1000

having

Grouping Data

1. Filtering With WHERE And **having**

Using the HAVING Clause

1. … Aggregate Functions With The **having** Clause
2. **having** AVG(list_price) > 1000
3. … AVG(list_price)FROM productGROUP BY category_id**having** AVG(list_price) > 1000
4. … so we will focus solely on the **having** clause.

Using the HAVING and WHERE Clauses Together

1. … model_year = 2016GROUP BY category_id**having** AVG(list_price) > 1000
2. **having** AVG(list_price) > 1000
3. … statement that includes both the **having** and WHERE clause in the same SQL statement.

in

The INSERT Clause With a Column List

1. The **in**SERT Clause With a Column List
2. Below is a basic example of an **in**SERT statement with a column list:
3. **in**SERT **in**TO city
4. Results of the **in**sert:
5. 1 USE world;2 **in**SERT **in**TO city 3 (name, countryCode,…

The INSERT Clause Without a Column List

1. The **in**SERT Clause Without a Column List
2. 1 USE world;2 **in**SERT **in**TO city 3 VALUES 4 (DEFAULT,…

Grouping Data

1. Filter**in**g With WHERE And HAV**in**G
2. Us**in**g the GROUP BY Clause

Improving the GROUP BY Query

1. Improv**in**g the GROUP BY Query
2. … List Price'FROM product p JO**in** category c ON p.category_id = c.category_idGROUP…
3. JO**in** category c

Using the HAVING Clause

1. Filter**in**g Aggregate Functions With The HAV**in**G Clause
2. HAV**in**G AVG(list_price) > 1000
3. … productGROUP BY category_idHAV**in**G AVG(list_price) > 1000
4. … previously discussed the preced**in**g l**in**es of code for this query so we will…

Using the HAVING and WHERE Clauses Together

1. … = 2016GROUP BY category_idHAV**in**G AVG(list_price) > 1000
2. HAV**in**G AVG(list_price) > 1000
3. … an example of a statement that **in**cludes both the HAV**in**G and WHERE clause…

Using the DISTINCT Statement

1. Removing Duplicate Values With DISTinCT
2. … COUNT(list_price), COUNT(DISTinCT list_price) FROM product;

The Subquery in an UPDATE statement

1. The Subquery in an UPDATE statement
2. … GNPOld = 0.00 3 WHERE Code in 4 (SELECT CountryCode FROM countrylanguage…
3. WHERE Code in

Create a Duplicate Table From An Existing Table

1. … Duplicate Table from an Existing Table with a Select Statement

The Subquery In a Delete Statement

1. The Subquery in a DELETE statement
2. … FROM city_bakWHERE CountryCode in (SELECT code FROM country …
3. WHERE CountryCode in
4. … uncheck “Safe Updates” checkbox in MySQL Preference. Please see below.

Benefits of Using Views

1. Benefits of Using Views
2. … country_nameFROM city ci JOin country co ON ci.CountryCode = co.Code;
3. JOin country co
4. Results by selecting from the city_country view:

Views That Allow UPDATE Statements

1. Creating Views That Can Be Used With an UPDATE Statement

Clustered vs. Non-clustered Indexes

1. Clustered vs. Non-clustered indexes

Aggregate Functions

1. SUM([DISTinCT] column_values)
2. Min([DISTinCT] column_values)
3. … average of the non-null columns in the expression
4. … AVG(list_price), SUM(list_price), Min(list_price), MAX(list_price),…
5. MAX([DISTinCT] column_values)
6. COUNT([DISTinCT] column_values)
7. … value of the non-null columns in the expression
8. numeric, date, string
9. AVG([DISTinCT] column_values)
10. … total of the non-null columns in the expression
11. numeric, date, string
12. … value off the non-null columns in the expression
13. … number of the non-null columns in the expression

The Subquery In a SELECT Statement

1. The Subquery **in** a SELECT Statement
2. WHERE CountryCode **in**
3. … city 4 WHERE CountryCode **in** 5 (SELECT code 6 …

Introduction

1. Before You Beg**in**
2. … databases that you can download and **in**stall **in** your local MySQL **in**stance.…
3. **in** a future edition, this book will **in**clude SQL design basics and guidance…

How to Retrieve Data From a Single Table

1. The **in** Keyword
2. DIST**in**CT Keyword
3. The clauses MUST appear **in** the order shown above.
4. Matches any s**in**gle character with**in** the given range.
5. … world;SELECT nameFROM countryWHERE name **in** ('Aruba', 'Barbados', 'Cuba',…
6. DIST**in**CT
7. Matches any s**in**gle character listed with**in** the brackets.
8. Match the pattern to the beg**in**n**in**g of the value be**in**g tested.
9. Let us break the statement l**in**e by l**in**e:
10. … but b must NOT be present to be **in**cluded
11. Elim**in**ates duplicate rows
12. Match any str**in**g of characters to the left of the symbol
13. SELECT name, **in**depYearFROM countryWHERE **in**depYear IS NULL;
14. Separates two str**in**g patterns and matches either one
15. Matches any s**in**gle character.
16. USE world;SELECT name, **in**depYearFROM countryWHERE name BETWEEN "Aruba" and…
17. … pattern to the end of the value be**in**g tested.
18. **in**teger Division
19. **in** the previous example, we created a new column that was a calculated value.…
20. Table 7. DIST**in**CT Keyword
21. … a and b are present, item is **in**cluded
22. SELECT DIST**in**CT cont**in**ent, nameFROM countryORDER BY cont**in**ent;
23. Match a s**in**gle character
24. … either a OR b is present item is **in**cluded
25. Modulo (rema**in**der)
26. We used the AS keyword then **in** quotes we put the new column alias of “People…

SQL Indexes Explained

1. When to Create an **in**dex
2. SQL **in**dexes

The Five Clauses of the SELECT Statement

1. The clauses MUST appear **in** the order shown above.
2. Let us break the statement l**in**e by l**in**e:

LIKE and REGEXP Operators

1. Matches any single character within the given range.
2. Matches any single character listed within the brackets.
3. Match the pattern to the beginning of the value being tested.
4. Match any string of characters to the left of the symbol
5. Separates two string patterns and matches either one
6. Matches any single character.
7. … pattern to the end of the value being tested.
8. Match a single character

Arithmetic Operators

1. integer Division
2. Modulo (remainder)

Column Aliases

1. in the previous example, we created a new column that was a calculated value.…
2. We used the AS keyword then in quotes we put the new column alias of “People…

IS NULL, BETWEEN, IN Operators

1. The in Keyword
2. … world;SELECT nameFROM countryWHERE name in ('Aruba', 'Barbados', 'Cuba',…
3. SELECT name, indepYearFROM countryWHERE indepYear IS NULL;
4. USE world;SELECT name, indepYearFROM countryWHERE name BETWEEN "Aruba" and…

AND, OR, NOT Logical Operators

1. … but b must NOT be present to be included
2. … a and b are present, item is included
3. … either a OR b is present item is included

DISTINCT Clause

1. DISTinCT Keyword
2. DISTinCT
3. Eliminates duplicate rows
4. Table 7. DISTinCT Keyword
5. SELECT DISTinCT continent, nameFROM countryORDER BY continent;

The JOIN Clause

1. The Join Clause
2. … write SQL statements more succinctly with an inner join clause using table…
3. JOin country co
4. Let us break the statement line by line:
5. … example of a SQL statement with an inner join clause using explicit syntax.
6. … FROM country 6 JOin city 5 ON city.CountryCode…
7. The results of the join query would yield the same results as shown below…
8. … FROM city ci 5 JOin country co 6 ON ci.CountryCode…

Joining More Than Two Tables

1. How to Jo**in** More than Two Tables
2. JO**in** countrylanguage cl.
3. … FROM city ci6 JO**in** country co 7 ON ci.CountryCode…

The OUTER JOIN Clause

1. The Outer Jo**in** Clause
2. … SQL statement with an outer jo**in** clause.
3. SELECT c.name, c.cont**in**ent, cl.language
4. FROM country c LEFT JO**in** countrylanguage cl
5. … world;2 SELECT c.name, c.cont**in**ent, cl.language3 FROM country c LEFT JO**in**…

How to Code a UNION

1. … populationFROM countryWHERE cont**in**ent = 'Oceania'
2. … population6 FROM country7 WHERE cont**in**ent = 'Oceania'8 ORDER BY name;

Date Functions

1. • Dates must be enclosed **in** quotes • You can pass a DATE or DATETIME…
2. * extracts the date from **in**put. If time is **in**cluded, the time is dropped.
3. USE world;SELECT name, cont**in**ent, DATE_FORMAT('2020-01-28', '%m/%d/%y')FROM…
4. M**in**utes, numeric (00..59)
5. … order_date, DATE_ADD(order_date, **in**TERVAL 1 DAY) AS 'ORDER DATE PLUS…
6. DATE_ADD(date, **in**terval expression unit)
7. DATE_ADD('2020-01-01', **in**TERVAL 1 DAY)
8. … DATETIME value equal to the orig**in**al value plus the specified **in**terval.

Numeric Functions

1. FLOOR, CEIL**in**G, TRUNCATE
2. … list_price, FLOOR(list_price), CEIL**in**G(list_price), TRUNCATE(list_price,…
3. CEIL**in**G(number)
4. CEIL**in**G(6.2)
5. Table 6. FLOOR, CEIL**in**G, TRUNCATE functions

String Functions

1. LOCATE, LENGTH, SUBSTR**in**G
2. str**in**g
3. SUBSTR**in**G(str,start[,length])
4. str**in**g
5. str**in**g
6. LEFT(str**in**g, num. characters)
7. TRIM(str**in**g)
8. str**in**g
9. … world;SELECT CONCAT(name, ', ', cont**in**ent)FROM country;
10. str**in**g
11. LTRIM(str**in**g)
12. str**in**g
13. str**in**g
14. RIGHT(str**in**g, num. characters)
15. … LENGTH('salmon'), SUBSTR**in**G('salmon',3,999);
16. Table 9. LOCATE. LENGTH, SUBSTR**in**G functions
17. LOCATE(f**in**d,search[,start])
18. str**in**g
19. LOWER(str**in**g)
20. SUBSTR**in**G('salmon',3,999)
21. str**in**g
22. str**in**g
23. LOCATE(), and LENGTH() accept a str**in**g but return an **in**teger. • SUBSTR**in**G()…
24. RTRIM(str**in**g)
25. str**in**g
26. UPPER(str**in**g)

index

SQL Indexes Explained

1. When to Create an **index**
2. SQL **index**es

Clustered vs. Non-clustered Indexes

1. Clustered vs. Non-clustered **index**es

indexes

Clustered vs. Non-clustered Indexes

1. Clustered vs. Non-clustered **indexes**

SQL Indexes Explained

1. SQL **indexes**

insert

The INSERT Clause With a Column List

1. The **insert** Clause With a Column List
2. Below is a basic example of an **insert** statement with a column list:
3. **insert** INTO city
4. Results of the **insert**:
5. 1 USE world;2 **insert** INTO city 3 (name, countryCode, district,…

The INSERT Clause Without a Column List

1. The **insert** Clause Without a Column List
2. 1 USE world;2 **insert** INTO city 3 VALUES 4 (DEFAULT,…

is null

How to Retrieve Data From a Single Table

1. **is null**
2. … IndepYearFROM countryWHERE IndepYear **is null**;

IS NULL, BETWEEN, IN Operators

1. **is null**
2. … IndepYearFROM countryWHERE IndepYear **is null**;

join

The JOIN Clause

1. The **join** Clause
2. … more succinctly with an inner **join** clause using table aliases. Instead of…
3. **join** country co
4. … a SQL statement with an inner **join** clause using explicit syntax.
5. … FROM country 6 **join** city 5 ON city.CountryCode…
6. The results of the **join** query would yield the same results as shown below whether…
7. … FROM city ci 5 **join** country co 6 ON ci.CountryCode…

Joining More Than Two Tables

1. How to **join** More than Two Tables
2. **join** countrylanguage cl.
3. … FROM city ci6 **join** country co 7 ON ci.CountryCode…

The OUTER JOIN Clause

1. The Outer **join** Clause
2. … a SQL statement with an outer **join** clause.
3. FROM country c LEFT **join** countrylanguage cl
4. … cl.language3 FROM country c LEFT **join** countrylanguage cl4 ON c.code = cl.CountryCode5…

Improving the GROUP BY Query

1. ... List Price'FROM product p **join** category c ON p.category_id = c.category_idGROUP...
2. **join** category c

Benefits of Using Views

1. ... country_nameFROM city ci **join** country co ON ci.CountryCode = co.Code;
2. **join** country co

left

String Functions

1. ... bike;SELECT category_name, **left**(category_name, 8) AS 'First 8 Characters', ...
2. **left**('Salmon', 3)

How to Retrieve Data From a Single Table

1. ... any string of characters to the **left** of the symbol

LIKE and REGEXP Operators

1. ... any string of characters to the **left** of the symbol

The OUTER JOIN Clause

1. FROM country c **left** JOIN countrylanguage cl
2. ... c.continent, cl.language3 FROM country c **left** JOIN countrylanguage cl4 ON...

String Functions

1. RIGHT, **left**
2. **left**(string, num. characters)
3. **left**('Salmon ')
4. SELECT LTRIM(' Salmon ') AS "**left** Trim", RTRIM(' Salmon ') AS...
5. Table 7. RIGHT, **left** functions

like

How to Retrieve Data From a Single Table

1. **like** and REGEXP Operators
2. **like** Symbol
3. Table 2. **like** Keyword
4. ... world;SELECT nameFROM countryWHERE name **like** 'A%'

LIKE and REGEXP Operators

1. **like** and REGEXP Operators
2. **like** Symbol
3. Table 2. **like** Keyword
4. ... world;SELECT nameFROM countryWHERE name **like** 'A%'

limit

How to Retrieve Data From a Single Table

1. **limit** 5;

The Five Clauses of the SELECT Statement

1. … “AFG”5 ORDER BY name6 **limit** 3
2. **limit** 5;

The Subquery In a SELECT Statement

1. … ORDER BY population 9 **limit** 5;

logical operators

How to Retrieve Data From a Single Table

1. AND, OR, NOT **logical operators**
2. Table 6. **logical operators**

AND, OR, NOT Logical Operators

1. AND, OR, NOT **logical operators**
2. Table 6. **logical operators**

ltrim

String Functions

1. TRIM, **ltrim**, RTRIM
2. **ltrim**(string)
3. SELECT **ltrim**(' Salmon ') AS "Left Trim", RTRIM(' Salmon ') AS "Right…

min

Aggregate Functions

1. **min**([DISTINCT] column_values)
2. … AVG(list_price), SUM(list_price), **min**(list_price), MAX(list_price),…

How to Retrieve Data From a Single Table

1. Eli**min**ates duplicate rows

DISTINCT Clause

1. Eli**min**ates duplicate rows

Date Functions

1. **min**utes, numeric (00..59)

now

How to Retrieve Data From a Single Table

1. … is that the column header is **now** population / SurfaceArea. However we can…

Column Aliases

1. … is that the column header is **now** population / SurfaceArea. However, we can…

Date Functions

1. **now**()
2. SELECT **now**() AS '**now**()', DATE('2020-01-01') AS 'DATE(), date only', …
3. **now**()

null

How to Retrieve Data From a Single Table

1. IS **null**
2. … IndepYearFROM countryWHERE IndepYear IS **null**;

IS NULL, BETWEEN, IN Operators

1. IS **null**
2. … IndepYearFROM countryWHERE IndepYear IS **null**;

Aggregate Functions

1. The average of the non-**null** columns in the expression
2. The highest value of the non-**null** columns in the expression
3. The total of the non-**null** columns in the expression
4. The lowest value off the non-**null** columns in the expression
5. The number of the non-**null** columns in the expression

or not

The JOIN Clause

1. … results as shown below whether **or not** table names are completely written out…

order by

How to Retrieve Data From a Single Table

1. … 'Barbados', 'Cuba', 'Bahamas')**order by** population ASC;
2. **order by** name
3. … 'caribbean'AND population > 100000**order by** population ASC;
4. … DISTINCT continent, nameFROM country**order by** continent;

The Five Clauses of the SELECT Statement

1. … CountryCode = “AFG”5 **order by** name6 LIMIT 3
2. **order by** name

IS NULL, BETWEEN, IN Operators

1. … 'Barbados', 'Cuba', 'Bahamas')**order by** population ASC;

AND, OR, NOT Logical Operators

1. … 'caribbean'AND population > 100000**order by** population ASC;

DISTINCT Clause

1. … DISTINCT continent, nameFROM country**order by** continent;

The OUTER JOIN Clause

1. … cl4 ON c.code = cl.CountryCode5 **order by** cl.language ASC;

How to Code a UNION

1. **order by** name;
2. … WHERE continent = 'Oceania'8 **order by** name;

Improving the GROUP BY Query

1. **order by** category_name;
2. … c.category_idGROUP BY category_name**order by** category_name;

The Subquery In a SELECT Statement

1. … region = 'Caribbean') 8 **order by** population 9 LIMIT 5;

outer join

The OUTER JOIN Clause

1. The **outer join** Clause
2. … snippet of a SQL statement with an **outer join** clause.

regexp

How to Retrieve Data From a Single Table

1. LIKE and **regexp** Operators
2. … world;SELECT nameFROM countryWHERE name **regexp** 'g[o,u]';
3. **regexp** Characters

LIKE and REGEXP Operators

1. LIKE and **regexp** Operators
2. … world;SELECT nameFROM countryWHERE name **regexp** 'g[o,u]';
3. **regexp** Characters

right

String Functions

1. **right**, LEFT
2. **right**(string, num. characters)
3. **right**('Salmon', 3)
4. **right**(' Salmon')
5. … RTRIM(' Salmon ') AS "**right** Trim", TRIM(' Salmon ') AS "Trim";
6. Table 7. **right**, LEFT functions
7. … AS 'First 8 Characters', **right**(category_name, 8) AS 'Last 8 Characters'FROM…

round

Improving the GROUP BY Query

1. … category_name, CONCAT('$', **round**(AVG(list_price),2)) AS 'Average List…
2. CONCAT('$', **round**(AVG(list_price),2)) AS 'Average List Price'

Numeric Functions

1. **round**
2. Table 5. **round** function
3. **round**(13.37, 1)
4. … world;SELECT name, LifeExpectancy, **round**(LifeExpectancy) FROM world.country;
5. **round**(number[, length])

rtrim

String Functions

1. TRIM, LTRIM, **rtrim**
2. … Salmon ') AS "Left Trim", **rtrim**(' Salmon ') AS "Right Trim", …
3. **rtrim**(string)

select

String Functions

1. **select** UPPER('Salmon'), LOWER('Salmon');
2. USE world;**select** CONCAT(name, ', ', continent)FROM country;
3. **select** FORMAT(list_price,2) FROM bike.product;
4. **select** LOCATE('al','salmon',1), LENGTH('salmon'), SUBSTRING('salmon',3,999);
5. **select** LTRIM(' Salmon ') AS "Left Trim", RTRIM(' Salmon ') AS "Right…
6. USE bike;**select** category_name, LEFT(category_name, 8) AS 'First 8 Characters', …

Simple GROUP BY Query

1. USE bike;**select** category_id, AVG(list_price)FROM productGROUP BY category_id
2. **select** category_id, AVG(list_price):

Improving the GROUP BY Query

1. USE bike;**select** category_name, CONCAT('$', ROUND(AVG(list_price),2)) AS…
2. **select** category_name,

Using the HAVING Clause

1. USE bike;**select** category_id, AVG(list_price)FROM productGROUP BY category_idHAVING…

Using the HAVING and WHERE Clauses Together

1. USE bike;**select** category_id, AVG(list_price)FROM productWHERE model_year = 2016GROUP…

COUNT(column_name) and COUNT(*)

1. USE bike;**select** COUNT(phone), COUNT(*) FROM CUSTOMER

Using the DISTINCT Statement

1. ExampleUSE bike;**select** COUNT(list_price), COUNT(DISTINCT list_price) FROM product;

The Subquery in an UPDATE statement

1. (**select** CountryCode FROM countrylanguage WHERE population = 0)
2. … 0.00 3 WHERE Code IN 4 (**select** CountryCode FROM countrylanguage…

Create a Duplicate Table From An Existing Table

1. … from an Existing Table with a **select** Statement
2. CREATE TABLE city_bak AS **select** * FROM city;
3. … CREATE TABLE city_bak AS **select** * FROM city;

The Subquery In a Delete Statement

1. (**select** code FROM country
2. … city_bakWHERE CountryCode IN (**select** code FROM country WHERE…

Benefits of Using Views

1. … WORLD;CREATE VIEW city_country AS**select** ci.name AS city_name, co.name AS country_nameFROM…
2. **select** ci.name AS city_name, co.name AS country_name
3. Results by **select**ing from the city_country view:

Aggregate Functions

1. USE bike;**select** AVG(list_price), SUM(list_price), MIN(list_price), MAX(list_price),…

The Subquery In a SELECT Statement

1. The Subquery in a **select** Statement
2. 1 USE world;2 **select** name, population 3 FROM city 4 WHERE…
3. **select** name, population
4. (**select** code

How to Retrieve Data From a Single Table

1. The Five Clauses of the **select** statement
2. USE world;**select** nameFROM countryWHERE name IN ('Aruba', 'Barbados', 'Cuba',...
3. **select** name, population / SurfaceArea AS “People per square mile”FROM...
4. **select** name, IndepYearFROM countryWHERE IndepYear IS NULL;
5. USE world;**select** nameFROM countryWHERE name REGEXP 'g[o,u]';
6. USE world;**select** name, populationFROM countryWHERE population > 1000000;
7. USE world;**select** name, populationFROM countryWHERE region = 'caribbean'AND population...
8. USE world;**select** name, population / SurfaceAreaAS "People per square mile"FROM...
9. USE world;**select** name, IndepYearFROM countryWHERE name BETWEEN "Aruba" and "Bahamas";
10. **select** name
11. **select** DISTINCT continent, nameFROM countryORDER BY continent;
12. USE world;**select** nameFROM countryWHERE name LIKE ‘A%’

The Five Clauses of the SELECT Statement

1. The Five Clauses of the **select** statement
2. ... Example:1 USE world;2 **select** name3 FROM city4 WHERE CountryCode...
3. **select** name

LIKE and REGEXP Operators

1. USE world;**select** nameFROM countryWHERE name REGEXP 'g[o,u]';
2. USE world;**select** nameFROM countryWHERE name LIKE ‘A%’

Arithmetic Operators

1. USE world;**select** name, population / SurfaceAreaAS "People per square mile"FROM...

Column Aliases

1. **select** name, population / SurfaceArea AS “People per square mile”FROM...

Comparison Operators

1. USE world;**select** name, populationFROM countryWHERE population > 1000000;

IS NULL, BETWEEN, IN Operators

1. USE world;**select** nameFROM countryWHERE name IN ('Aruba', 'Barbados', 'Cuba',...
2. **select** name, IndepYearFROM countryWHERE IndepYear IS NULL;
3. USE world;**select** name, IndepYearFROM countryWHERE name BETWEEN "Aruba" and "Bahamas";

AND, OR, NOT Logical Operators

1. USE world;**select** name, populationFROM countryWHERE region = 'caribbean'AND population...

DISTINCT Clause

1. **select** DISTINCT continent, nameFROM countryORDER BY continent;

The JOIN Clause

1. **select** ci.name AS “City Name”, co.name AS “Country Name”
2. 1 USE world;2 **select** city.name AS "City Name", 3 country.name…
3. … clause and referenced in the **select** and ON clause:
4. 1 USE world;2 **select** ci.name AS "City Name", 3 co.name…

Joining More Than Two Tables

1. 1 USE world; 2 **select** ci.name AS "City Name",3 co.name…

The OUTER JOIN Clause

1. **select** c.name, c.continent, cl.language
2. 1 USE world;2 **select** c.name, c.continent, cl.language3 FROM country c LEFT JOIN…

How to Code a UNION

1. **select** name, populationFROM countryWHERE continent = 'Oceania'
2. **select** name, populationFROM cityWHERE CountryCode = 'AUS'
3. 1 USE world;2 **select** name, population3 FROM city WHERE CountryCode = 'AUS'4 UNION5…

Date Functions

1. **select** NOW() AS 'NOW()', DATE('2020-01-01') AS 'DATE(), date only', CURRENT_DATE…
2. **select** DATEDIFF('2018-01-01', '2019-01-01') AS 'Date Difference';
3. USE world;**select** name, continent, DATE_FORMAT('2020-01-28', '%m/%d/%y')FROM country;
4. USE bike;**select** order_date, DATE_ADD(order_date, INTERVAL 1 DAY) AS 'ORDER…

Numeric Functions

1. USE bike;**select** list_price, FLOOR(list_price), CEILING(list_price), TRUNCATE(list_price,…
2. USE world;**select** name, LifeExpectancy, ROUND(LifeExpectancy) FROM world.country;

sql indexes

SQL Indexes Explained

1. **sql indexes**

sql view

SQL View Explained

1. **sql view**s

sql views

SQL View Explained

1. **sql views**

subquery

The Subquery in an UPDATE statement

1. The **subquery** in an UPDATE statement

The Subquery In a Delete Statement

1. The **subquery** in a DELETE statement

The Subquery In a SELECT Statement

1. The **subquery** in a SELECT Statement

sum

Aggregate Functions

1. **sum**([DISTINCT] column_values)
2. … bike;SELECT AVG(list_price), **sum**(list_price), MIN(list_price), MAX(list_price),…

trim

String Functions

1. **trim**, L**trim**, R**trim**
2. **trim**(string)
3. **trim**(‘ Salmon ‘)
4. L**trim**(string)
5. Table 8. **trim** functions
6. SELECT L**trim**(' Salmon ') AS "Left **trim**", R**trim**(' Salmon ') AS…
7. R**trim**(string)

truncate

Numeric Functions

1. FLOOR, CEILING, **truncate**
2. **truncate**(7.9)
3. … FLOOR(list_price), CEILING(list_price), **truncate**(list_price, 0)FROM product;
4. Table 6. FLOOR, CEILING, **truncate** functions
5. **truncate**(NUMBER, length)

union

How to Code a UNION

1. How to Code a **union**
2. **union**
3. … city WHERE CountryCode = 'AUS'4 **union**5 SELECT name, population6 FROM country7…

update

The UPDATE Clause With a Column List

1. The **update** Clause
2. **update** city
3. 1 USE world; 2 **update** city 3 SET Population = 65000, district…

The Subquery in an UPDATE statement

1. The Subquery in an **update** statement
2. 1 **update** country 2 SET GNPOld = 0.00 3 WHERE Code IN 4 (SELECT…
3. **update** country

The Subquery In a Delete Statement

1. … Before you can run a DELETE or **update** statement without a WHERE clause, you…

Views That Allow UPDATE Statements

1. … Views That Can Be Used With an **update** Statement

utc_date

Date Functions

1. … CURRENT_TIME AS 'CURRENT_TIME', **utc_date** AS '**utc_date**', UTC_TIME AS…
2. **utc_date**()
3. **utc_date**

utc_time

Date Functions

1. **utc_time**
2. **utc_time**()
3. … UTC_DATE AS 'UTC_DATE', **utc_time** AS '**utc_time**';

view

Benefits of Using Views

1. Benefits of Using **view**s
2. USE WORLD;CREATE **view** city_country ASSELECT ci.name AS city_name, co.name AS…
3. CREATE **view** city_country AS
4. … selecting from the city_country **view**:

Views That Allow UPDATE Statements

1. Creating **view**s That Can Be Used With an UPDATE Statement

SQL View Explained

1. SQL **view**s

views

Benefits of Using Views

1. Benefits of Using **views**

Views That Allow UPDATE Statements

1. Creating **views** That Can Be Used With an UPDATE Statement

SQL View Explained

1. SQL **views**

where

The DELETE Clause

1. … DELETE 3 FROM city 4 **where** name = 'san felipe' AND countrycode…
2. **where** name = 'san felipe' AND countrycode = 'chl';

Grouping Data

1. Filtering With **where** And HAVING

Using the HAVING and WHERE Clauses Together

1. **where** model_year = 2016
2. … category_id, AVG(list_price)FROM product**where** model_year = 2016GROUP BY category_idHAVING…
3. … includes both the HAVING and **where** clause in the same SQL statement.

The Subquery in an UPDATE statement

1. … CountryCode FROM countrylanguage **where** population = 0)
2. … SET GNPOld = 0.00 3 **where** Code IN 4 (SELECT CountryCode FROM…
3. **where** Code IN

The Subquery In a Delete Statement

1. USE world;DELETE FROM city_bak**where** CountryCode IN (SELECT code FROM country …
2. **where** region = 'Central Africa');
3. **where** CountryCode IN
4. … or UPDATE statement without a **where** clause, you must uncheck “Safe Updates”…

The Subquery In a SELECT Statement

1. **where** CountryCode IN
2. … population 3 FROM city 4 **where** CountryCode IN 5 (SELECT…
3. **where** region = 'Caribbean')

How to Retrieve Data From a Single Table

1. … world;SELECT nameFROM country**where** name IN ('Aruba', 'Barbados', 'Cuba', 'Bahamas')ORDER…
2. SELECT name, IndepYearFROM country**where** IndepYear IS NULL;
3. … world;SELECT nameFROM country**where** name REGEXP 'g[o,u]';
4. … world;SELECT name, populationFROM country**where** population > 1000000;
5. … world;SELECT name, populationFROM country**where** region = 'caribbean'AND population…
6. … world;SELECT name, IndepYearFROM country**where** name BETWEEN "Aruba" and "Bahamas";
7. … world;SELECT nameFROM country**where** name LIKE 'A%'

The Five Clauses of the SELECT Statement

1. … SELECT name3 FROM city4 **where** CountryCode = "AFG"5 ORDER…

LIKE and REGEXP Operators

1. … world;SELECT nameFROM country**where** name REGEXP 'g[o,u]';
2. … world;SELECT nameFROM country**where** name LIKE 'A%'

Comparison Operators

1. … world;SELECT name, populationFROM country**where** population > 1000000;

IS NULL, BETWEEN, IN Operators

1. … world;SELECT nameFROM country**where** name IN ('Aruba', 'Barbados', 'Cuba', 'Bahamas')ORDER…
2. SELECT name, IndepYearFROM country**where** IndepYear IS NULL;
3. … world;SELECT name, IndepYearFROM country**where** name BETWEEN "Aruba" and "Bahamas";

AND, OR, NOT Logical Operators

1. … world;SELECT name, populationFROM country**where** region = 'caribbean'AND population…

How to Code a UNION

1. SELECT name, populationFROM country**where** continent = 'Oceania'
2. SELECT name, populationFROM city**where** CountryCode = 'AUS'
3. … SELECT name, population3 FROM city **where** CountryCode = 'AUS'4 UNION5 SELECT…

Date Functions

1. Year for the week **where** Sunday is the first day of the week, numeric, four digits;…
2. Week (01..53), **where** Sunday is the first day of the week; WEEK() mode 2; used…
3. Week (00..53), **where** Sunday is the first day of the week; WEEK() mode 0
4. Week (00..53), **where** Monday is the first day of the week; WEEK() mode 1
5. Week (01..53), **where** Monday is the first day of the week; WEEK() mode 3; used…
6. Year for the week, **where** Monday is the first day of the week, numeric, four…