Filtreleme

Bazı durumlarda, özellikle language gibi küçük tablolar için bir tablodaki tüm satırları almak isteyebilirsiniz. Bununla birlikte, çoğu zaman, bir tablodan her satırı almak istemeyeceksiniz, ancak ilgilenilmeyen satırları filtrelemenin bir yolunu isteyeceksiniz. Where yan cümlesi bunun için vardır. Where yan tümcesi, istenmeyen satırları sonuç kümenizden filtrelemek için kullanılan mekanizmadır.

Örneğin, belki bir film kiralamakla ilgileniyorsunuz, ancak yalnızca en az bir hafta saklanabilecek G dereceli filmlerle ilgileniyorsunuz. Aşağıdaki sorgu, yalnızca bu kriterleri karşılayan filmleri almak için bir where yan tümcesi kullanır:

```
mysql> SELECT title
    -> FROM film
    -> WHERE rating = 'G' AND rental_duration >= 7;
+-------------------------+
| title                   |
+-------------------------+
| BLANKET BEVERLY         |
| BORROWERS BEDAZZLED     |
| BRIDE INTRIGUE          |
| CATCH AMISTAD           |
| CITIZEN SHREK           |
| COLDBLOODED DARLING     |
| CONTROL ANTHEM          |
| CRUELTY UNFORGIVEN      |
| DARN FORRESTER          |
| DESPERATE TRAINSPOTTING |
| DIARY PANIC             |
| DRACULA CRYSTAL         |
| EMPIRE MALKOVICH        |
| FIREHOUSE VIETNAM       |
| GILBERT PELICAN         |
| GRADUATE LORD           |
| GREASE YOUTH            |
| GUN BONNIE              |
| HOOK CHARIOTS           |
| MARRIED GO              |
| MENAGERIE RUSHMORE      |
| MUSCLE BRIGHT           |
| OPERATION OPERATION     |
| PRIMARY GLASS           |
| REBEL AIRPORT           |
| SPIKING ELEMENT         |
| TRUMAN CRAZY            |
| WAKE JAWS               |
| WAR NOTTING             |
+-------------------------+
29 rows in set (0.00 sec)
```

Kaynak: Alan Beaulieu, 2020, Learning SQL, 3rd Edition, O'Reilly.

Bu durumda, where yan tümcesi film tablosundaki 1000 satırın 971'ini filtrelemiştir. Bu, yan tümce iki filtre koşulu içerir, ancak gerektiği kadar çok koşul ekleyebilirsiniz; bireysel koşullar, and, or, not gibi operatörler kullanılarak ayrılır.
İki koşulu birbirinden ayıran operatörü veya olarak değiştirirseniz ne olacağını görelim:

```
mysql> SELECT title
    -> FROM film
    -> WHERE rating = 'G' OR rental_duration >= 7;
+----------------------------+
| title                      |
+----------------------------+
| ACE GOLDFINGER             |
| ADAPTATION HOLES           |
| AFFAIR PREJUDICE           |
| AFRICAN EGG                |
| ALAMO VIDEOTAPE            |
| AMISTAD MIDSUMMER          |
| ANGELS LIFE                |
| ANNIE IDENTITY             |
|...                         |
| WATERSHIP FRONTIER         |
| WEREWOLF LOLA              |
| WEST LION                  |
| WESTWARD SEABISCUIT        |
| WOLVES DESIRE              |
| WON DARES                  |
| WORKER TARZAN              |
| YOUNG LANGUAGE             |
+----------------------------+
340 rows in set (0.00 sec)
```

Koşulları ve operatörü kullanarak ayırdığınızda, sonuç kümesine dahil edilmek için tüm koşulların true olarak değerlendirilmesi gerekir; veya kullandığınızda bir satırın dahil edilebilmesi için koşullardan yalnızca birinin doğru olarak değerlendirilmesi gerekir, bu da sonuç kümesinin boyutunun neden 29'dan 340 satıra sıçradığını açıklar.

Peki, where yan tümcenizde hem and and or operatörlerini kullanmanız gerekiyorsa ne yapmalısınız? Koşulları birlikte gruplamak için parantez kullanmalısınız. Sonraki sorgu, yalnızca G olarak derecelendirilen ve 7 veya daha fazla gün için mevcut olan veya PG-13 olarak derecelendirilen ve 3 veya daha az gün için mevcut olan filmlerin sonuç kümesine dahil edileceğini belirtir:

```
mysql> SELECT title, rating, rental_duration
    -> FROM film
    -> WHERE (rating = 'G' AND rental_duration >= 7)
    ->   OR (rating = 'PG-13' AND rental_duration < 4);

+-------------------------+--------+-----------------+
| title                   | rating | rental_duration |
+-------------------------+--------+-----------------+
| ALABAMA DEVIL           | PG-13  |               3 |
| BACKLASH UNDEFEATED     | PG-13  |               3 |
| BILKO ANONYMOUS         | PG-13  |               3 |
| BLANKET BEVERLY         | G      |               7 |
| BORROWERS BEDAZZLED     | G      |               7 |
| BRIDE INTRIGUE          | G      |               7 |
| CASPER DRAGONFLY        | PG-13  |               3 |
| CATCH AMISTAD           | G      |               7 |
| CITIZEN SHREK           | G      |               7 |
| COLDBLOODED DARLING     | G      |               7 |
|...                                                 |
| TREASURE COMMAND        | PG-13  |               3 |
| TRUMAN CRAZY            | G      |               7 |
| WAIT CIDER              | PG-13  |               3 |
| WAKE JAWS               | G      |               7 |
| WAR NOTTING             | G      |               7 |
| WORLD LEATHERNECKS      | PG-13  |               3 |
+-------------------------+--------+-----------------+
68 rows in set (0.00 sec)
```

Farklı operatörleri karıştırırken koşul gruplarını ayırmak için her zaman parantez kullanmalısınız, böylece siz, veritabanı sunucusu ve daha sonra kodunuzu değiştirmek için gelen herkes aynı yorumda bulunabilir.

Group by ve Having

Şimdiye kadarki tüm sorgular, herhangi bir manipülasyon olmadan ham verileri aldı. Ancak bazen, sonuç kümenizi almadan önce veri tabanı sunucusunun verileri biraz olgunlaşmasını gerektirecek verilerinizde eğilimleri bulmak isteyeceksiniz. Bu tür bir mekanizma, verileri sütun değerlerine göre gruplamak için kullanılan **group by** yan cümlesi ile gerçekleştirilir. Örneğin, 40 veya daha fazla film kiralamış tüm müşterileri bulmak istediğinizi varsayalım. Kiralama tablosundaki 16.044 satırın tümüne

bakmak yerine, sunucuya tüm kiralamaları müşteriye göre gruplandırmasını, her müşteri için kiralama sayısını saymasını ve ardından yalnızca kiralama sayısı en az 40 olan müşterileri iade etmesini söyleyen bir sorgu yazabilirsiniz. Satır grupları oluşturmak için **group by** yan tümcesini kullanırken, aynı şekilde, where yan tümcesinin ham verileri filtrelemenize izin verdiği şekilde gruplanmış verileri filtrelemenize izin veren **having** yan tümcesini de kullanabilirsiniz.

```
mysql> SELECT c.first_name, c.last_name, count(*)
    -> FROM customer c
    ->   INNER JOIN rental r
    ->   ON c.customer_id = r.customer_id
    -> GROUP BY c.first_name, c.last_name
    -> HAVING count(*) >= 40;
+------------+-----------+----------+
| first_name | last_name | count(*) |
+------------+-----------+----------+
| TAMMY      | SANDERS   |       41 |
| CLARA      | SHAW      |       42 |
| ELEANOR    | HUNT      |       46 |
| SUE        | PETERS    |       40 |
| MARCIA     | DEAN      |       42 |
| WESLEY     | BULL      |       40 |
| KARL       | SEAL      |       45 |
+------------+-----------+----------+
7 rows in set (0.03 sec)
```

Order By

Genel olarak, bir sorgudan döndürülen sonuç kümesindeki satırlar belirli bir sırada değildir. Sonuç kümenizin sıralanmasını istiyorsanız, sunucuya **order by** yan tümcesini kullanarak sonuçları sıralamasını söylemeniz gerekir:

Örneğin, 14 Haziran 2005'te film kiralayan tüm müşterileri döndüren önceki bir sorguya başka bir bakış:

```
mysql> SELECT c.first_name, c.last_name,
    ->   time(r.rental_date) rental_time
    -> FROM customer c
    ->   INNER JOIN rental r
    ->   ON c.customer_id = r.customer_id
    -> WHERE date(r.rental_date) = '2005-06-14'
    -> ORDER BY c.last_name;
+------------+-----------+-------------+
| first_name | last_name | rental_time |
+------------+-----------+-------------+
| DANIEL     | CABRAL    | 23:09:38    |
| CATHERINE  | CAMPBELL  | 23:17:03    |
| HERMAN     | DEVORE    | 23:35:09    |
| AMBER      | DIXON     | 23:42:56    |
| JOYCE      | EDWARDS   | 23:16:26    |
| JEANETTE   | GREENE    | 23:54:46    |
| SONIA      | GREGORY   | 23:50:11    |
| TERRENCE   | GUNDERSON | 23:47:35    |
| CHARLES    | KOWALSKI  | 23:54:34    |
| MATTHEW    | MAHAN     | 23:25:58    |
| GWENDOLYN  | MAY       | 23:16:27    |
| MIRIAM     | MCKINNEY  | 23:07:08    |
| ELMER      | NOE       | 22:55:13    |
| JEFFERY    | PINSON    | 22:53:33    |
| MINNIE     | ROMERO    | 23:00:34    |
| TERRANCE   | ROUSH     | 23:12:46    |
+------------+-----------+-------------+
16 rows in set (0.01 sec)
```

Bu örnekte durum böyle olmasa da, büyük müşteri listeleri genellikle aynı soyadına sahip birden fazla kişiyi içerecektir, bu nedenle sıralama kriterlerini kişinin adını da içerecek şekilde genişletmek isteyebilirsiniz.
Bunu, last_name sütunundan sonra first_name sütununu yan tümce sırasına göre ekleyerek gerçekleştirebilirsiniz:

```
mysql> SELECT c.first_name, c.last_name,
    ->   time(r.rental_date) rental_time
    -> FROM customer c
    ->   INNER JOIN rental r
    ->   ON c.customer_id = r.customer_id
    -> WHERE date(r.rental_date) = '2005-06-14'
    -> ORDER BY c.last_name, c.first_name;
+------------+-----------+-------------+
| first_name | last_name | rental_time |
+------------+-----------+-------------+
| DANIEL     | CABRAL    | 23:09:38    |
| CATHERINE  | CAMPBELL  | 23:17:03    |
| HERMAN     | DEVORE    | 23:35:09    |
| AMBER      | DIXON     | 23:42:56    |
| JOYCE      | EDWARDS   | 23:16:26    |
| JEANETTE   | GREENE    | 23:54:46    |
| SONIA      | GREGORY   | 23:50:11    |
| TERRENCE   | GUNDERSON | 23:47:35    |
| CHARLES    | KOWALSKI  | 23:54:34    |
| MATTHEW    | MAHAN     | 23:25:58    |
| GWENDOLYN  | MAY       | 23:16:27    |
| MIRIAM     | MCKINNEY  | 23:07:08    |
| ELMER      | NOE       | 22:55:13    |
| JEFFERY    | PINSON    | 22:53:33    |
| MINNIE     | ROMERO    | 23:00:34    |
| TERRANCE   | ROUSH     | 23:12:46    |
+------------+-----------+-------------+
16 rows in set (0.01 sec)
```

Birden fazla sütun eklediğinizde, sütunların sırasına göre yan tümcenizde göründüğü sıra bir fark yaratır. Eğer iki sütunun sırasını yan tümceye göre değiştirecek olsaydınız, sonuç kümesinde ilk olarak Amber Dixon görünürdü.

Ascend ve Descend

Sıralama yaparken, **ascending** veya **descending** anahtar sözcükleri aracılığıyla artan veya azalan sıralama belirleme seçeneğiniz vardır. Varsayılan sıralama artandır, bu nedenle azalan bir sıralama kullanmak istiyorsanız **desc** anahtar sözcüğünü eklemeniz gerekir. Örneğin, aşağıdaki sorgu, 14 Haziran 2005'te film kiralayan tüm müşterileri azalan kiralama süresine göre gösterir:

```
mysql> SELECT c.first_name, c.last_name,
    ->   time(r.rental_date) rental_time
    -> FROM customer c
    ->   INNER JOIN rental r
    ->   ON c.customer_id = r.customer_id
    -> WHERE date(r.rental_date) = '2005-06-14'
    -> ORDER BY time(r.rental_date) desc;
+------------+-----------+-------------+
| first_name | last_name | rental_time |
+------------+-----------+-------------+
| JEANETTE   | GREENE    | 23:54:46    |
| CHARLES    | KOWALSKI  | 23:54:34    |
| SONIA      | GREGORY   | 23:50:11    |
| TERRENCE   | GUNDERSON | 23:47:35    |
| AMBER      | DIXON     | 23:42:56    |
| HERMAN     | DEVORE    | 23:35:09    |
| MATTHEW    | MAHAN     | 23:25:58    |
| CATHERINE  | CAMPBELL  | 23:17:03    |
| GWENDOLYN  | MAY       | 23:16:27    |
| JOYCE      | EDWARDS   | 23:16:26    |
| TERRANCE   | ROUSH     | 23:12:46    |
| DANIEL     | CABRAL    | 23:09:38    |
| MIRIAM     | MCKINNEY  | 23:07:08    |
| MINNIE     | ROMERO    | 23:00:34    |
| ELMER      | NOE       | 22:55:13    |
| JEFFERY    | PINSON    | 22:53:33    |
+------------+-----------+-------------+
16 rows in set (0.01 sec)
```

Sayısal Yer Tutucularla Sıralama

Select yan tümcenizdeki sütunları kullanarak sıralama yapıyorsanız, sütunlara ad yerine select yan tümcesindeki konumlarına göre başvuruda bulunmayı seçebilirsiniz. Bu, özellikle önceki örnekte olduğu gibi bir ifadeye göre sıralama yapıyorsanız yardımcı olabilir.

```
mysql> SELECT c.first_name, c.last_name,
    ->   time(r.rental_date) rental_time
    -> FROM customer c
    ->   INNER JOIN rental r
    ->   ON c.customer_id = r.customer_id
    -> WHERE date(r.rental_date) = '2005-06-14'
    -> ORDER BY 3 desc;
+------------+-----------+-------------+
| first_name | last_name | rental_time |
+------------+-----------+-------------+
| JEANETTE   | GREENE    | 23:54:46    |
| CHARLES    | KOWALSKI  | 23:54:34    |
| SONIA      | GREGORY   | 23:50:11    |
| TERRENCE   | GUNDERSON | 23:47:35    |
| AMBER      | DIXON     | 23:42:56    |
| HERMAN     | DEVORE    | 23:35:09    |
| MATTHEW    | MAHAN     | 23:25:58    |
| CATHERINE  | CAMPBELL  | 23:17:03    |
| GWENDOLYN  | MAY       | 23:16:27    |
| JOYCE      | EDWARDS   | 23:16:26    |
| TERRANCE   | ROUSH     | 23:12:46    |
| DANIEL     | CABRAL    | 23:09:38    |
| MIRIAM     | MCKINNEY  | 23:07:08    |
| MINNIE     | ROMERO    | 23:00:34    |
| ELMER      | NOE       | 22:55:13    |
| JEFFERY    | PINSON    | 22:53:33    |
+------------+-----------+-------------+
16 rows in set (0.01 sec)
```

Koşullar

Bir where yan tümcesi **and** ve **or** operatörleri ile ayrılmış bir veya daha fazla koşul içerebilir. Birden çok koşul yalnızca **and** operatörü tarafından ayrılırsa, satırın sonuç kümesine dahil edilmesi için tüm koşulların doğru olarak değerlendirilmesi gerekir.

```
WHERE first_name = 'STEVEN' AND create_date > '2006-01-01'
```

Bu iki koşul göz önüne alındığında, yalnızca adı Steven olan ve oluşturma tarihi 1 Ocak 2006'dan sonra olan satırlar sonuç kümesine dahil edilecektir. Bu örnek yalnızca iki koşul kullanıyor olsa da, where yan tümcenizde kaç koşul olursa olsun, bunlar and operatörüyle ayrılmışlarsa, satırın sonuç kümesine dahil edilebilmesi için hepsinin doğru olarak değerlendirilmesi gerekir.

Bununla birlikte, where yan tümcesindeki tüm koşullar **or** operatörü tarafından ayrılırsa, satırın sonuç kümesine dahil edilebilmesi için koşullardan yalnızca birinin doğru olarak değerlendirilmesi gerekir.

```
WHERE first_name = 'STEVEN' OR create_date > '2006-01-01'
```

Belirli bir satırın sonuç kümesine dahil edilmesinin üç farklı yol vardır:
- Adı Steven ve oluşturma tarihi 1 Ocak 2006'dan sonra olanlar.
- Adı Steven ve oluşturulma tarihi 1 Ocak 2006'da veya daha önce olanlar.
- Adı Steven'dan farklı ama oluşturma tarihi 1 Ocak 2006'dan sonra.

Intermediate result	Final result
WHERE true OR true	true
WHERE true OR false	true
WHERE false OR true	true
WHERE false OR false	false

Parantezler

Where yan tümceniz hem **and** hem de **or** operatörlerini kullanan üç veya daha fazla koşul içeriyorsa, niyetinizi hem veritabanı sunucusuna hem de kodunuzu okuyan başka birine daha açık belli etmek için parantez kullanmalısınız.

Not Operatörü

Intermediate result	Final result
WHERE (true OR true) AND true	true
WHERE (true OR false) AND true	true
WHERE (false OR true) AND true	true
WHERE (false OR false) AND true	false
WHERE (true OR true) AND false	false
WHERE (true OR false) AND false	false
WHERE (false OR true) AND false	false
WHERE (false OR false) AND false	false

Kaydı 1 Ocak 2006'dan sonra oluşturulan adı Steven veya soyadı Young olan

```
WHERE NOT (first_name = 'STEVEN' OR last_name = 'YOUNG')
  AND create_date > '2006-01-01'
```

kişileri aramak yerine, yalnızca adın Steven olmadığı veya soyadı Young olmayan ve 1 Ocak 2006'dan sonra oluşturulmuş satırlar alınır.

```
WHERE (first_name = 'STEVEN' OR last_name = 'YOUNG')
  AND create_date > '2006-01-01'
```

Intermediate result	Final result
WHERE NOT (true OR true) AND true	false
WHERE NOT (true OR false) AND true	false
WHERE NOT (false OR true) AND true	false
WHERE NOT (false OR false) AND true	true
WHERE NOT (true OR true) AND false	false
WHERE NOT (true OR false) AND false	false
WHERE NOT (false OR true) AND false	false
WHERE NOT (false OR false) AND false	false

Veritabanı sunucusunun işlemesi kolay olsa da, bir kişinin not operatörünü içeren bir where yan tümcesini değerlendirmesi genellikle zordur.

```
WHERE first_name <> 'STEVEN' AND last_name <> 'YOUNG'
  AND create_date > '2006-01-01'
```

Kaynak: Alan Beaulieu, 2020, Learning SQL, 3rd Edition, O'Reilly.

Koşul Tipleri

Yazdığınız veya karşılaştığınız filtre koşullarının büyük bir yüzdesi, aşağıdaki gibi column = expression formunda olacaktır:

```
title = 'RIVER OUTLAW'
fed_id = '111-11-1111'
amount = 375.25
film_id = (SELECT film_id FROM film WHERE title = 'RIVER OUTLAW')
```

Bu tür koşullar, bir ifadeyi diğerine eşitledikleri için eşitlik koşulları olarak adlandırılır. İlk üç örnek, bir sütunu sabit değere (iki dize ve bir sayı) eşitler ve dördüncü örnek, bir sütunu bir alt sorgudan döndürülen değere eşitler. Aşağıdaki sorgu, biri on yan tümcesinde (bir birleştirme koşulu) ve diğeri where yan tümcesinde (bir filtre koşulu) olmak üzere iki eşitlik koşulu kullanır:

```
mysql> SELECT c.email
    -> FROM customer c
    ->   INNER JOIN rental r
    ->   ON c.customer_id = r.customer_id
    -> WHERE date(r.rental_date) = '2005-06-14';
+-----------------------------------------+
| email                                   |
+-----------------------------------------+
| CATHERINE.CAMPBELL@sakilacustomer.org |
| JOYCE.EDWARDS@sakilacustomer.org      |
| AMBER.DIXON@sakilacustomer.org        |
| JEANETTE.GREENE@sakilacustomer.org    |
| MINNIE.ROMERO@sakilacustomer.org      |
| GWENDOLYN.MAY@sakilacustomer.org      |
| SONIA.GREGORY@sakilacustomer.org      |
| MIRIAM.MCKINNEY@sakilacustomer.org    |
| CHARLES.KOWALSKI@sakilacustomer.org   |
| DANIEL.CABRAL@sakilacustomer.org      |
| MATTHEW.MAHAN@sakilacustomer.org      |
| JEFFERY.PINSON@sakilacustomer.org     |
| HERMAN.DEVORE@sakilacustomer.org      |
| ELMER.NOE@sakilacustomer.org          |
| TERRANCE.ROUSH@sakilacustomer.org     |
| TERRENCE.GUNDERSON@sakilacustomer.org |
+-----------------------------------------+
16 rows in set (0.03 sec)
```

Oldukça yaygın bir başka koşul türü, iki ifadenin eşit olmadığını iddia eden eşitsizlik koşuludur.

```
mysql> SELECT c.email
    -> FROM customer c
    ->   INNER JOIN rental r
    ->   ON c.customer_id = r.customer_id
    -> WHERE date(r.rental_date) <> '2005-06-14';

+-----------------------------------+
| email                             |
+-----------------------------------+
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
| MARY.SMITH@sakilacustomer.org     |
...
| AUSTIN.CINTRON@sakilacustomer.org |
| AUSTIN.CINTRON@sakilacustomer.org |
| AUSTIN.CINTRON@sakilacustomer.org |
| AUSTIN.CINTRON@sakilacustomer.org |
| AUSTIN.CINTRON@sakilacustomer.org |
| AUSTIN.CINTRON@sakilacustomer.org |
| AUSTIN.CINTRON@sakilacustomer.org |
| AUSTIN.CINTRON@sakilacustomer.org |
+-----------------------------------+
16028 rows in set (0.03 sec)
```

Eşitlik/eşitsizlik koşulları, veriler değiştirilirken yaygın olarak kullanılır. Örneğin film kiralama şirketinin yılda bir kez eski hesap satırlarını kaldırma politikası olduğunu varsayalım. Göreviniz, kiralama tarihinin 2004 olduğu kiralama tablosundaki satırları kaldırmaktır.

```
DELETE FROM rental
WHERE year(rental_date) = 2004;
```

Bir ifadenin başka bir ifadeye eşit olduğunu (veya eşit olmadığını) kontrol etmenin yanı sıra, bir ifadenin belirli bir aralığa düşüp düşmediğini kontrol eden koşullar oluşturabilirsiniz.

```
mysql> SELECT customer_id, rental_date
    -> FROM rental
    -> WHERE rental_date < '2005-05-25';
+-------------+---------------------+
| customer_id | rental_date         |
+-------------+---------------------+
|         130 | 2005-05-24 22:53:30 |
|         459 | 2005-05-24 22:54:33 |
|         408 | 2005-05-24 23:03:39 |
|         333 | 2005-05-24 23:04:41 |
|         222 | 2005-05-24 23:05:21 |
|         549 | 2005-05-24 23:08:07 |
|         269 | 2005-05-24 23:11:53 |
|         239 | 2005-05-24 23:31:46 |
+-------------+---------------------+
8 rows in set (0.00 sec)


mysql> SELECT customer_id, rental_date
    -> FROM rental
    -> WHERE rental_date <= '2005-06-16'
    ->   AND rental_date >= '2005-06-14';
+-------------+---------------------+
| customer_id | rental_date         |
+-------------+---------------------+
|         416 | 2005-06-14 22:53:33 |
|         516 | 2005-06-14 22:55:13 |
|         239 | 2005-06-14 23:00:34 |
|         285 | 2005-06-14 23:07:08 |
|         310 | 2005-06-14 23:09:38 |
|         592 | 2005-06-14 23:12:46 |
...
|         148 | 2005-06-15 23:20:26 |
|         237 | 2005-06-15 23:36:37 |
|         155 | 2005-06-15 23:55:27 |
|         341 | 2005-06-15 23:57:20 |
|         149 | 2005-06-15 23:58:53 |
+-------------+---------------------+
364 rows in set (0.00 sec)
```

Between

Menziliniz için hem üst hem de alt sınırınız olduğunda, iki ayrı koşul kullanmak yerine, operatörler arasını kullanan tek bir koşul kullanmayı seçebilirsiniz:

```
mysql> SELECT customer_id, rental_date
    -> FROM rental
    -> WHERE rental_date BETWEEN '2005-06-14' AND '2005-06-16';
+-------------+---------------------+
| customer_id | rental_date         |
+-------------+---------------------+
|         416 | 2005-06-14 22:53:33 |
|         516 | 2005-06-14 22:55:13 |
|         239 | 2005-06-14 23:00:34 |
|         285 | 2005-06-14 23:07:08 |
|         310 | 2005-06-14 23:09:38 |
|         592 | 2005-06-14 23:12:46 |
...
|         148 | 2005-06-15 23:20:26 |
|         237 | 2005-06-15 23:36:37 |
|         155 | 2005-06-15 23:55:27 |
|         341 | 2005-06-15 23:57:20 |
|         149 | 2005-06-15 23:58:53 |
+-------------+---------------------+
364 rows in set (0.00 sec)
```

Between operatörünü kullanırken akılda tutulması gereken birkaç şey vardır. Her zaman önce aralığın alt sınırını ve sonra aralığın üst sınırını belirtmelisiniz. Önce üst sınırı yanlışlıkla belirtirseniz şunlar olur:

```
mysql> SELECT customer_id, rental_date
    -> FROM rental
    -> WHERE rental_date BETWEEN '2005-06-16' AND '2005-06-14';
Empty set (0.00 sec)
```

Gördüğünüz gibi, hiçbir veri döndürülmez. Bunun nedeni, sunucunun gerçekte, aşağıdaki gibi <= ve >= operatörlerini kullanarak tek koşulunuzdan iki koşul oluşturmasıdır:

```
SELECT customer_id, rental_date
    -> FROM rental
    -> WHERE rental_date >= '2005-06-16'
    ->   AND rental_date <= '2005-06-14'
Empty set (0.00 sec)
```

Kaynak: Alan Beaulieu, 2020, Learning SQL, 3rd Edition, O'Reilly.

```
mysql> SELECT customer_id, payment_date, amount
    -> FROM payment
    -> WHERE amount BETWEEN 10.0 AND 11.99;
+-------------+---------------------+--------+
| customer_id | payment_date        | amount |
+-------------+---------------------+--------+
|           2 | 2005-07-30 13:47:43 |  10.99 |
|           3 | 2005-07-27 20:23:12 |  10.99 |
|          12 | 2005-08-01 06:50:26 |  10.99 |
|          13 | 2005-07-29 22:37:41 |  11.99 |
|          21 | 2005-06-21 01:04:35 |  10.99 |
|          29 | 2005-07-09 21:55:19 |  10.99 |
...
|         571 | 2005-06-20 08:15:27 |  10.99 |
|         572 | 2005-06-17 04:05:12 |  10.99 |
|         573 | 2005-07-31 12:14:19 |  10.99 |
|         591 | 2005-07-07 20:45:51 |  11.99 |
|         592 | 2005-07-06 22:58:31 |  11.99 |
|         595 | 2005-07-31 11:51:46 |  10.99 |
+-------------+---------------------+--------+
114 rows in set (0.01 sec)
```

Tarih ve sayı aralıklarının anlaşılması kolay olsa da, görselleştirilmesi biraz daha zor olan String dizeleri arayan koşullar da oluşturabilirsiniz. Örneğin, soyadı belirli bir aralıkta olan müşterileri aradığınızı varsayalım. Soyadı FA ve FR arasında kalan müşterileri döndüren bir sorgu:

```
mysql> SELECT last_name, first_name
    -> FROM customer
    -> WHERE last_name BETWEEN 'FA' AND 'FR';
+-----------+------------+
| last_name | first_name |
+-----------+------------+
| FARNSWORTH | JOHN       |
| FENNELL    | ALEXANDER  |
| FERGUSON   | BERTHA     |
| FERNANDEZ  | MELINDA    |
| FIELDS     | VICKI      |
| FISHER     | CINDY      |
| FLEMING    | MYRTLE     |
| FLETCHER   | MAE        |
| FLORES     | JULIA      |
| FORD       | CRYSTAL    |
| FORMAN     | MICHEAL    |
| FORSYTHE   | ENRIQUE    |
| FORTIER    | RAUL       |
| FORTNER    | HOWARD     |
| FOSTER     | PHYLLIS    |
| FOUST      | JACK       |
| FOWLER     | JO         |
| FOX        | HOLLY      |
+------------+------------+
18 rows in set (0.00 sec)
```

Soyadı FR ile başlayan beş müşteri varken, FRANKLIN gibi bir isim aralığın dışında olduğu için sonuçlara dahil edilmedi. Ancak, sağ taraftaki aralığı FRB'ye genişleterek beş müşteriden dördünü alabiliriz:

```
mysql> SELECT last_name, first_name
    -> FROM customer
    -> WHERE last_name BETWEEN 'FA' AND 'FRB';
+------------+------------+
| last_name  | first_name |
+------------+------------+
| FARNSWORTH | JOHN       |
| FENNELL    | ALEXANDER  |
| FERGUSON   | BERTHA     |
| FERNANDEZ  | MELINDA    |
| FIELDS     | VICKI      |
| FISHER     | CINDY      |
| FLEMING    | MYRTLE     |
| FLETCHER   | MAE        |
| FLORES     | JULIA      |
| FORD       | CRYSTAL    |
| FORMAN     | MICHEAL    |
| FORSYTHE   | ENRIQUE    |
| FORTIER    | RAUL       |
| FORTNER    | HOWARD     |
| FOSTER     | PHYLLIS    |
| FOUST      | JACK       |
| FOWLER     | JO         |
| FOX        | HOLLY      |
| FRALEY     | JUAN       |
| FRANCISCO  | JOEL       |
| FRANKLIN   | BETH       |
| FRAZIER    | GLENDA     |
+------------+------------+
22 rows in set (0.00 sec)
```

Bazı durumlarda, bir ifadeyi tek bir değerle veya değer aralığıyla değil, sonlu bir değerler kümesiyle sınırlamış olursunuz. Örneğin, 'G' veya 'PG' derecelendirmesine sahip tüm filmleri bulmak isteyebilirsiniz:

```
mysql> SELECT title, rating
    -> FROM film
    -> WHERE rating = 'G' OR rating = 'PG';
+---------------------------+--------+
| title                     | rating |
+---------------------------+--------+
| ACADEMY DINOSAUR          | PG     |
| ACE GOLDFINGER            | G      |
| AFFAIR PREJUDICE          | G      |
| AFRICAN EGG               | G      |
| AGENT TRUMAN              | PG     |
| ALAMO VIDEOTAPE           | G      |
| ALASKA PHANTOM            | PG     |
| ALI FOREVER               | PG     |
| AMADEUS HOLY              | PG     |
...
| WEDDING APOLLO            | PG     |
| WEREWOLF LOLA             | G      |
| WEST LION                 | G      |
| WIZARD COLDBLOODED        | PG     |
| WON DARES                 | PG     |
| WONDERLAND CHRISTMAS      | PG     |
| WORDS HUNTER              | PG     |
| WORST BANGER              | PG     |
| YOUNG LANGUAGE            | G      |
+---------------------------+--------+
372 rows in set (0.00 sec)
```

Bu, yan tümcenin (iki koşul veya birlikte) oluşturulması mümkün olma da ifadeler de kümesinin 10 veya 20 durum içerdiğini hayal edin. Bu durumlar için bunun yerine in operatörünü kullanabilirsiniz:

```
SELECT title, rating
FROM film
WHERE rating IN ('G','PG');
```

in operatörü ile kümede kaç ifade olursa olsun tek bir koşul yazabilirsiniz.

('G', 'PG' gibi) gibi kendi ifade kümenizi yazmanın yanı sıra, sizin için bir küme oluşturmak için bir alt sorgu kullanabilirsiniz. Örneğin, başlığı 'PET' dizesini içeren herhangi bir filmin aile izlemesi için güvenli olacağını varsayarsanız, bu filmlerle ilişkili tüm derecelendirmeleri almak için film tablosuna karşı bir alt sorgu yürütebilirsiniz.

```
mysql> SELECT title, rating
    -> FROM film
    -> WHERE rating IN (SELECT rating FROM film WHERE title LIKE '%PET%');
+--------------------------+--------+
| title                    | rating |
+--------------------------+--------+
| ACADEMY DINOSAUR         | PG     |
| ACE GOLDFINGER           | G      |
| AFFAIR PREJUDICE         | G      |
| AFRICAN EGG              | G      |
| AGENT TRUMAN             | PG     |
| ALAMO VIDEOTAPE          | G      |
| ALASKA PHANTOM           | PG     |
| ALI FOREVER              | PG     |
| AMADEUS HOLY             | PG     |
...
| WEDDING APOLLO           | PG     |
| WEREWOLF LOLA            | G      |
| WEST LION                | G      |
| WIZARD COLDBLOODED       | PG     |
| WON DARES                | PG     |
| WONDERLAND CHRISTMAS     | PG     |
| WORDS HUNTER             | PG     |
| WORST BANGER             | PG     |
| YOUNG LANGUAGE           | G      |
+--------------------------+--------+
372 rows in set (0.00 sec)
```

Bazen bir ifade kümesi içinde belirli bir ifadenin var olup olmadığını görmek istersiniz.

```
SELECT title, rating
FROM film
WHERE rating NOT IN ('PG-13','R', 'NC-17');
```

Bu sorgu, önceki sorgularla aynı 372 satır kümesini döndürecek olan 'PG-13' ,'R' veya 'NC-17' olarak derecelendirilmemiş tüm hesapları bulur.

Şimdiye kadar, tam bir dizeyi, bir String diziyi veya bir String diziyi tanımlayan koşullarla karşılaşıldı; son koşul türü, kısmi dize eşleşmeleriyle ilgilidir. Örneğin, soyadı Q ile başlayan tüm müşterileri bulmak isteyebilirsiniz. Aşağıdaki gibi, soyadı sütununun ilk harfini çıkarmak için yerleşik bir işlev kullanabilirsiniz:

```
mysql> SELECT last_name, first_name
    -> FROM customer
    -> WHERE left(last_name, 1) = 'Q';
+-------------+------------+
| last_name   | first_name |
+-------------+------------+
| QUALLS      | STEPHEN    |
| QUINTANILLA | ROGER      |
| QUIGLEY     | TROY       |
+-------------+------------+
3 rows in set (0.00 sec)
```

Yerleşik left() işlevi görevini yaparken size fazla esneklik sağlamaz. Bunun yerine, bir sonraki bölümde gösterildiği gibi arama ifadeleri oluşturmak için joker karakterler kullanabilirsiniz.
Kısmi dize eşleşmelerini ararken:
• Belirli bir karakterle başlayan/biten diziler
• Bir alt diziyle başlayan/biten diziler
• Dize içinde herhangi bir yerde belirli bir karakter içeren dizeler
• Dize içinde herhangi bir yerde bir alt dize içeren dizeler.

Wildcard character	Matches
_	Exactly one character
%	Any number of characters (including 0)

• Alt çizgi karakteri tek bir karakterin yerini alırken, yüzde işareti değişken sayıda karakterin yerini alabilir. Arama ifadelerini kullanan koşullar oluştururken, aşağıdaki gibi like operatörünü kullanırsınız:

```
mysql> SELECT last_name, first_name
    -> FROM customer
    -> WHERE last_name LIKE '_A_T%S';
+-----------+------------+
| last_name | first_name |
+-----------+------------+
| MATTHEWS  | ERICA      |
| WALTERS   | CASSANDRA  |
| WATTS     | SHELLY     |
+-----------+------------+
3 rows in set (0.00 sec)
```

Kaynak: Alan Beaulieu, 2020, Learning SQL, 3rd Edition, O'Reilly.

Önceki örnekteki arama ifadesi, ikinci konumda bir A ve dördüncü konumda bir T içeren, ardından herhangi bir sayıda karakter gelen ve S ile biten dizeleri belirtir.

Search expression	Interpretation
F%	Strings beginning with *F*
%t	Strings ending with *t*
%bas%	Strings containing the substring 'bas'
_ _t_	Four-character strings with a *t* in the third position
_ _ _-_ _-_ _ _ _	11-character strings with dashes in the fourth and seventh positions

```
mysql> SELECT last_name, first_name
    -> FROM customer
    -> WHERE last_name LIKE 'Q%' OR last_name LIKE 'Y%';
+-------------+------------+
| last_name   | first_name |
+-------------+------------+
| QUALLS      | STEPHEN    |
| QUIGLEY     | TROY       |
| QUINTANILLA | ROGER      |
| YANEZ       | LUIS       |
| YEE         | MARVIN     |
| YOUNG       | CYNTHIA    |
+-------------+------------+
6 rows in set (0.00 sec)
```

NULL

Tablodaki bir sütun için henüz belirlenmiş bir değer yoksa, eklenecek olan değer uygulanabilir değilse veya eklenecek olan sütun değeri şimdilik bilinmiyorsa null değerler kullanılır. Null ile çalışırken şunları hatırlamanız gerekir:

- Bir ifade null olabilir, ancak asla null ‘a eşit olamaz.
- İki null değer asla birbirine eşit değildir.
- Bir ifadenin boş olup olmadığını test etmek için, aşağıda gösterildiği gibi, **is null** operatörünü kullanmanız gerekir:

```
mysql> SELECT rental_id, customer_id
    -> FROM rental
    -> WHERE return_date IS NULL;
+-----------+-------------+
| rental_id | customer_id |
+-----------+-------------+
|     11496 |         155 |
|     11541 |         335 |
|     11563 |          83 |
|     11577 |         219 |
|     11593 |          99 |
...
|     15867 |         505 |
|     15875 |          41 |
|     15894 |         168 |
|     15966 |         374 |
+-----------+-------------+
183 rows in set (0.01 sec)
```

Bu sorgu, asla iade edilmeyen tüm film kiralamalarını bulur. is null yerine = null kullanan aynı sorgu:

```
mysql> SELECT rental_id, customer_id
    -> FROM rental
    -> WHERE return_date = NULL;
Empty set (0.01 sec)
```

Bir sütuna değer atanıp atanmadığını görmek istiyorsanız, is not null operatörünü aşağıdaki gibi kullanabilirsiniz:

```
mysql> SELECT rental_id, customer_id, return_date
    -> FROM rental
    -> WHERE return_date IS NOT NULL;
+-----------+-------------+---------------------+
| rental_id | customer_id | return_date         |
+-----------+-------------+---------------------+
|         1 |         130 | 2005-05-26 22:04:30 |
|         2 |         459 | 2005-05-28 19:40:33 |
|         3 |         408 | 2005-06-01 22:12:39 |
|         4 |         333 | 2005-06-03 01:43:41 |
|         5 |         222 | 2005-06-02 04:33:21 |
|         6 |         549 | 2005-05-27 01:32:07 |
|         7 |         269 | 2005-05-29 20:34:53 |
...
```

```
|     16043 |         526 | 2005-08-31 03:09:03 |
|     16044 |         468 | 2005-08-25 04:08:39 |
|     16045 |          14 | 2005-08-25 23:54:26 |
|     16046 |          74 | 2005-08-27 18:02:47 |
|     16047 |         114 | 2005-08-25 02:48:48 |
|     16048 |         103 | 2005-08-31 21:33:07 |
|     16049 |         393 | 2005-08-30 01:01:12 |
+-----------+-------------+---------------------+
15861 rows in set (0.02 sec)
```

Mayıs-Ağustos 2005 arasında iade edilmeyen tüm kiralamaları bulmanız istendiğini varsayalım.

```
mysql> SELECT rental_id, customer_id, return_date
    -> FROM rental
    -> WHERE return_date NOT BETWEEN '2005-05-01' AND '2005-09-01';
+-----------+-------------+---------------------+
| rental_id | customer_id | return_date         |
+-----------+-------------+---------------------+
|     15365 |         327 | 2005-09-01 03:14:17 |
|     15388 |          50 | 2005-09-01 03:50:23 |
|     15392 |         410 | 2005-09-01 01:14:15 |
|     15401 |         103 | 2005-09-01 03:44:10 |
|     15415 |         204 | 2005-09-01 02:05:56 |
...
|     15977 |         550 | 2005-09-01 22:12:10 |
|     15982 |         370 | 2005-09-01 21:51:31 |
|     16005 |         466 | 2005-09-02 02:35:22 |
|     16020 |         311 | 2005-09-01 18:17:33 |
|     16033 |         226 | 2005-09-01 02:36:15 |
|     16037 |          45 | 2005-09-01 02:48:04 |
|     16040 |         195 | 2005-09-02 02:19:33 |
+-----------+-------------+---------------------+
62 rows in set (0.01 sec)
```

Bu 62 kiralamanın Mayıs-Ağustos aranlığının dışında gerçekleştiği doğru olsa da, verilere dikkatlice bakarsanız, geri dönen tüm satırların boş olmayan bir dönüş tarihi olduğunu göreceksiniz. Fakat peki ya asla geri dönmeyen 183 kiralık? Biri, bu 183 satırın Mayıs ve Ağustos ayları arasında da iade edilmediğini iddia edebilir, böylece sonuç kümesine de dahil edilmelidir. Soruyu doğru şekilde cevaplamak için, bazı satırların RETURN_DATE sütununda bir null içerme olasılığını hesaba katmanız gerekir:

```
mysql> SELECT rental_id, customer_id, return_date
    -> FROM rental
    -> WHERE return_date IS NULL
    ->   OR return_date NOT BETWEEN '2005-05-01' AND '2005-09-01';
+-----------+-------------+---------------------+
| rental_id | customer_id | return_date         |
+-----------+-------------+---------------------+
|     11496 |         155 | NULL                |
|     11541 |         335 | NULL                |
|     11563 |          83 | NULL                |
|     11577 |         219 | NULL                |
|     11593 |          99 | NULL                |
...
|     15939 |         382 | 2005-09-01 17:25:21 |
|     15942 |         210 | 2005-09-01 18:39:40 |
|     15966 |         374 | NULL                |
|     15971 |         187 | 2005-09-02 01:28:33 |
|     15973 |         343 | 2005-09-01 20:08:41 |
|     15977 |         550 | 2005-09-01 22:12:10 |
|     15982 |         370 | 2005-09-01 21:51:31 |
|     16005 |         466 | 2005-09-02 02:35:22 |
|     16020 |         311 | 2005-09-01 18:17:33 |
|     16033 |         226 | 2005-09-01 02:36:15 |
|     16037 |          45 | 2005-09-01 02:48:04 |
|     16040 |         195 | 2005-09-02 02:19:33 |
+-----------+-------------+---------------------+
245 rows in set (0.01 sec)
```

Kaynak: Alan Beaulieu, 2020, Learning SQL, 3rd Edition, O’Reilly.