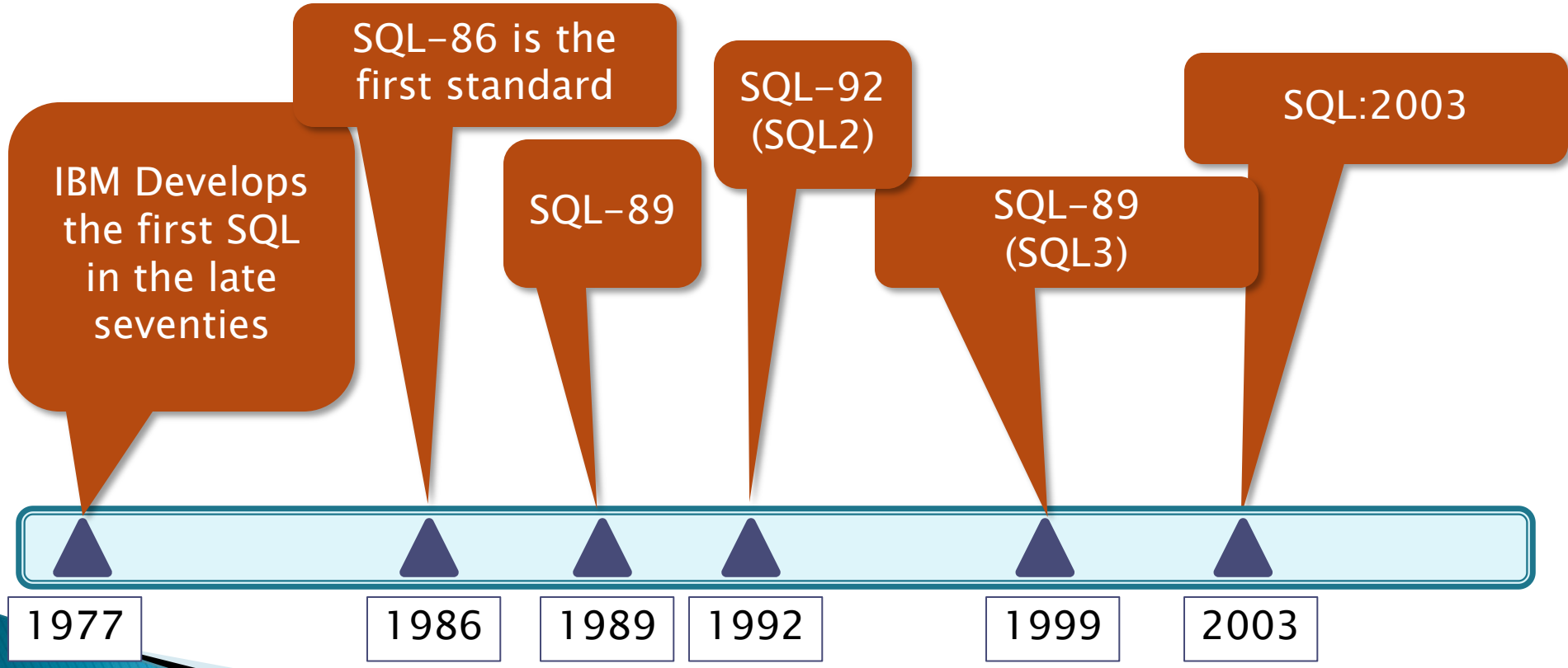
Structured Query Language

EHSDI

eBuzima

The Structured Query Language



Queries

- SELECT
- Used to retrieve information from a database

- Syntax:

```
select colum_name_a, column_name_b from table_name
```

Queries

```
select * from forecast_data
```

my_date	description	details	id
2009-05-06	warm	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	warm	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	cold	NULL	5
2009-05-06	cold	NULL	6
2009-05-06	cold	NULL	7

7 rows in set (0.00 sec)

Queries

```
select my_date, id from forecast_data
```

my_date	id
2009-05-06	1
2009-05-06	2
2009-05-06	3
2009-05-06	4
2009-05-06	5
2009-05-06	6
2009-05-06	7

DISTINCT

- The `distinct` keyword returns only unique occurrences of a particular field

- Syntax:

```
select distinct column_a, column_b from table_name
```

DISTINCT

```
+------------+-------------+---------+----+
| my_date    | description | details | id |
+------------+-------------+---------+----+
| 2009-05-06 | warm        | NULL    |  1 |
| 2009-05-06 | warm        | NULL    |  2 |
| 2009-05-06 | warm        | NULL    |  3 |
| 2009-05-06 | cold        | NULL    |  4 |
| 2009-05-06 | cold        | NULL    |  5 |
| 2009-05-06 | cold        | NULL    |  6 |
| 2009-05-06 | cold        | NULL    |  7 |
+------------+-------------+---------+----+
7 rows in set (0.00 sec)
```

```
mysql> select distinct description from forecast_data;
+-------------+
| description |
+-------------+
| warm        |
| cold        |
+-------------+
2 rows in set (0.00 sec)
```

WHERE

- Used to retrieve records that fit a criteria defined in the where clause

```
select column_a, column_b from table_name where column_name
operator value
```

WHERE

my_date	description	details	id
2009-05-06	warm	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	warm	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	cold	NULL	5
2009-05-06	cold	NULL	6
2009-05-06	cold	NULL	7

7 rows in set (0.00 sec)

```
select * from forecast_data where id > 4;
```

my_date	description	details	id
2009-05-06	cold	NULL	5
2009-05-06	cold	NULL	6
2009-05-06	cold	NULL	7

Operators in the WHERE Clause

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	If you know the exact value you want to return for at least one of the columns

source:
http://www.w3schools.com/SQl/sql_where.asp

AND ...OR

my_date	description	details	id
2009-05-06	warm	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	warm	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	cold	NULL	5
2009-05-06	cold	NULL	6
2009-05-06	cold	NULL	7

7 rows in set (0.00 sec)

```
select * from forecast_data where id > 6 OR description='cold';
```

my_date	description	details	id
2009-05-06	cold	NULL	4
2009-05-06	cold	NULL	5
2009-05-06	cold	NULL	6
2009-05-06	cold	NULL	7

AND..OR

my_date	description	details	id
2009-05-06	warm	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	warm	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	cold	NULL	5
2009-05-06	cold	NULL	6
2009-05-06	cold	NULL	7

7 rows in set (0.00 sec)

```
select * from forecast_data where id > 6 AND
 description='cold'or details is not null;
```

my_date	description	details	id
2009-05-06	cold	NULL	7

ORDER BY

- `order by` is used to order a list of records retreived

```
select column_a, column_b order by column_a,column_b asc|desc
```

ORDER BY

```
mysql> select * from forecast_data;
+------------+--------------+----------+----+
| my_date    | description  | details  | id |
+------------+--------------+----------+----+
| 2009-05-06 | very hot     | NULL     |  1 |
| 2009-05-06 | warm         | NULL     |  2 |
| 2009-05-06 | mild         | NULL     |  3 |
| 2009-05-06 | cold         | NULL     |  4 |
| 2009-05-06 | very cold    | NULL     |  5 |
| 2009-05-06 | mild         | NULL     |  6 |
| 2009-05-06 | hot          | NULL     |  7 |
+------------+--------------+----------+----+
7 rows in set (0.00 sec)
```

```
select description from forecast_data asc
```

```
+--------------+
| description  |
+--------------+
| very hot     |
| warm         |
| mild         |
| cold         |
| very cold    |
| mild         |
| hot          |
+--------------+
```

INSERT

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7

7 rows in set (0.00 sec)

```
insert into forecast_data  values (curdate(), 'hot', NULL,8);
```

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8

INSERT Into Specified Columns

```
insert into forecast_data (my_date,id) values (curdate(),9);
```

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-25	NULL	NULL	9

UPDATE

- Modify existing records within a table.

```
update table_name set column_1 = value, column_2=value where
column_3 = x
```

UPDATE

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-25	NULL	NULL	9

9 rows in set (0.00 sec)

```
update forecast_data set description='warm', my_date=curdate()
where id = 9
```

```
mysql> select * from forecast_data;
```

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	warm	NULL	9

DELETE

```
delete from tablename where column_a = value
```

DELETE

- Delete records from a table. May be used with a where clause.

```
delete from tablename where column_a = value
```

DELETE

```
mysql> select * from forecast_data;
+-------------+--------------+----------+----+
| my_date     | description  | details  | id |
+-------------+--------------+----------+----+
| 2009-05-06  | very hot     | NULL     |  1 |
| 2009-05-06  | warm         | NULL     |  2 |
| 2009-05-06  | mild         | NULL     |  3 |
| 2009-05-06  | cold         | NULL     |  4 |
| 2009-05-06  | very cold    | NULL     |  5 |
| 2009-05-06  | mild         | NULL     |  6 |
| 2009-05-06  | hot          | NULL     |  7 |
| 2009-06-25  | hot          | NULL     |  8 |
| 2009-06-26  | warm         | NULL     |  9 |
+-------------+--------------+----------+----+
```

```
delete from forecast_data where id = 9
```

```
mysql> select * from forecast_data;
+-------------+--------------+----------+----+
| my_date     | description  | details  | id |
+-------------+--------------+----------+----+
| 2009-05-06  | very hot     | NULL     |  1 |
| 2009-05-06  | warm         | NULL     |  2 |
| 2009-05-06  | mild         | NULL     |  3 |
| 2009-05-06  | cold         | NULL     |  4 |
| 2009-05-06  | very cold    | NULL     |  5 |
| 2009-05-06  | mild         | NULL     |  6 |
| 2009-05-06  | hot          | NULL     |  7 |
| 2009-06-25  | hot          | NULL     |  8 |
+-------------+--------------+----------+----+
8 rows in set (0.00 sec)
```

A nice page to try a few queries

- http://www.w3schools.com/SQl/sql_tryit.asp

More SQL

LIMIT

- Limit the number of records returned in a query

```
LIMIT {[offset,] row_count | row_count OFFSET offset}
```

LIMIT

```
sql> select * from forecast_data order by
+------------+--------------+---------+----+
| my_date    | description  | details | id |
+------------+--------------+---------+----+
| 2009-05-06 | very hot     | NULL    |  1 |
| 2009-05-06 | warm         | NULL    |  2 |
| 2009-05-06 | mild         | NULL    |  3 |
| 2009-05-06 | cold         | NULL    |  4 |
| 2009-05-06 | very cold    | NULL    |  5 |
| 2009-05-06 | mild         | NULL    |  6 |
| 2009-05-06 | hot          | NULL    |  7 |
| 2009-06-25 | hot          | NULL    |
+------------+--------------+-----
```

rowcount

```
select id from forecast_data order by id limit 5,1
```

offset

```
+----+
| id |
+----+
|  6 |
+----+
```

- Limit the number of records returned in a query

LIKE and NOT LIKE

- Select records having a description field with anything after the word "very"

```
select * from forecast_data where description like 'very%'
```

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	very cold	NULL	5

LIKE and NOT LIKE

```
select * from forecast_data where description like '%cold'
```

my_date	description	details	id
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5

LIKE and NOT LIKE

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

```
select * from forecast_data where description NOT LIKE '%il%'
```

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

Single Character Wildcard '_'

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

```
select * from forecast_data where description LIKE 'v_ry c_ld'
```

my_date	description	details	id
2009-05-06	very cold	NULL	5

REGEXP

- A very cool and powerful capability in MySQL and other databases is the ability to incorporate regular expression syntax when selecting data.

REGEXP

. match any character
? match zero or one
* match zero or more
+ match one or more
{n} match n times
{m,n} match m through n times
{n,} match n or more times
^ beginning of line
$ end of line

REGEXP

- [[:<:]] match beginning of words
  [[:>:]] match ending of words
  [:class:] match a character class
  i.e., [:alpha:] for letters
  [:space:] for whitespace
  [:punct:] for punctuation
  [:upper:] for upper case letters
  [abc] match one of enclosed chars
  [^xyz] match any char not enclosed
  | separates alternatives

REGEXP

- MySQL interprets a backslash (\) character as an escape character. To use a backslash in a regular expression, you must escape it with another backslash (\\).

REGEXP

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

```
select * from forecast_data where description regexp '^[mh].*'
```

my_date	description	details	id
2009-05-06	mild	NULL	3
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8

COUNT(*)

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

```
select count(*) from forecast_data where description regexp '^[mh].*'
```

```
mysql> select count(*) f
+----------+
| count(*) |
+----------+
|        4 |
+----------+
1 row in set (0.12 sec)
```

MAX and MIN

- Retrieves the maximum or minimum from a specific table or set of records (numbers, string(varchar), and dates)

```
mysql> select max(my_date) from forecast_data;
+---------------+
| max(my_date)  |
+---------------+
| 2009-06-26    |
+---------------+
1 row in set (0.05 sec)

mysql> select max(id) from forecast_data;
+----------+
| max(id)  |
+----------+
|        9 |
+----------+
```

COUNT(*) ar

- Collect records i

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

```
select count(*) from forecast_data group by description
```

count(*)	description
1	cold
2	hot
2	mild
1	very
1	very cold
1	very hot
1	warm

EHSDI
eBuzima

IN

- Specify multiple values in a where clause. Equivalent to using multiple OR

```
Select column_a from table_name where column_a in (value_a, value_b)
```

IN

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

```
select * from forecast_data where id in (1,4,6)
```

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	cold	NULL	4
2009-05-06	mild	NULL	6

BETWEEN

- Select records within a range of two values (dates, strings (varchar), numbers)

```
select column_a from table_name where column_a between value1 and
value2;
```

BETWEEN

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

9 rows in set (0.02 sec)

```
select * from forecast_data where my_date between '2009-05-07' and curdate();
```

my_date	description	details	id
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9

Alias

- An alias can be used for columns or tables
- This makes queries easier to read, shorter

An alias for a table

```
select column_name from table_name as alias_name
```

An alias for a column

```
select column_name as alias_name from table_name
```

Table Alias

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9
2009-06-28	cold	NULL	10

```
select f.my_date from forecast_data as f
```

my_date
2009-05-06
2009-05-06
2009-05-06
2009-05-06
2009-05-06
2009-05-06
2009-05-06
2009-06-25
2009-06-26
2009-06-28

10 rows in set (0.00 sec)

Join

- Used to query data from multiple tables using relationships between fields in each table
  - Inner Join
  - Left Join
  - Right Join
  - Full Join

Two Tables

forecast_data table

my_date	description	details	id
2009-05-06	very hot	NULL	1
2009-05-06	warm	NULL	2
2009-05-06	mild	NULL	3
2009-05-06	cold	NULL	4
2009-05-06	very cold	NULL	5
2009-05-06	mild	NULL	6
2009-05-06	hot	NULL	7
2009-06-25	hot	NULL	8
2009-06-26	very	NULL	9
2009-06-28	cold	NULL	10

crime table

id	date	crime
1	2009-06-28	armed robbery
2	2008-05-05	car theft

Inner Join

- Select records which have a match in both tables. Records without a match in the corresponding table will not be shown

```
select column_name from table_a inner join table_b on table_a.id = table_b.id_val
```

my_date	description	details	id	id	date	crime
2009-06-28	cold	NULL	10	1	2009-06-28	armed robbery

Inner Join

```
select forecast_data.description,crime.crime
from
fore                                    ime
on f                                    crime.date;
```

```
mysql> select * from forecast_data;
+------------+-------------+---------+----+
| my_date    | description | details | id |
+------------+-------------+---------+----+
| 2009-05-06 | very hot    | NULL    |  1 |
| 2009-05-06 | warm        | NULL    |  2 |
| 2009-05-06 | mild        | NULL    |  3 |
| 2009-05-06 | cold        | NULL    |  4 |
| 2009-05-06 | very cold   | NULL    |  5 |
| 2009-05-06 | mild        | NULL    |  6 |
| 2009-05-06 | hot         | NULL    |  7 |
| 2009-06-25 | hot         | NULL    |  8 |
| 2009-06-26 | very        | NULL    |  9 |
| 2009-06-28 | cold        | NULL    | 10 |
+------------+-------------+---------+----+
```

```
+---
| description | `crime`
+---------------+-----------------+
| cold          | armed robbery   |
+---------------+-----------------+
```

```
+------+------------+---------------+
| id   | date       | crime         |
+------+------------+---------------+
|    1 | 2009-06-28 | armed robbery |
|    2 | 2008-05-05 | car theft     |
+------+------------+---------------+
```

EHSDI
eBuzima

Inner Join Alternate Syntax (same effect)

```
select
forecast_data.description,crime.crime
from
forecast_data , crime
where
forecast_data.my_date = crime.date;
```

Left Join

- Return all records from the left table, even if there are no matches.

```
select
forecast_data.description,crime.crime from
forecast_data left join crime
on forecast_data.my_date = crime.date;
```

description	crime
very hot	NULL
warm	NULL
mild	NULL
cold	NULL
very cold	NULL
mild	NULL
hot	NULL
hot	NULL
very	NULL
cold	armed robbery

10 rows in set (0.00 sec)

Right Join

- Return all the records from the right table even if there are no matching records

```
select
forecast_data.description,crime.crime
from forecast_data right join crime
on forecast_data.my_date = crime.date;
```

description	crime
cold	armed robbery
NULL	car theft

Adding Where Clause to Joins

```
select * from forecast_data left join
crime
on forecast_data.my_date = crime.date
where forecast_data.description = 'cold'
```

my_date	description	details	id	id	date	crime
2009-05-06	cold	NULL	4	NULL	NULL	NULL
2009-06-28	cold	NULL	10	1	2009-06-28	armed robbery

Adding a Third Table

- The Suspect table

given_name	family_name	id
Rowan	Seymour	1
Bob	Jones	2
Amy	Tang	3

Updated Crime Table

- Now there is a suspect id column

```
mysql> select * from crime;
+------+------------+----------------+------------+
| id   | date       | crime          | suspect_id |
+------+------------+----------------+------------+
|    1 | 2009-06-28 | armed robbery  |          1 |
|    2 | 2008-05-05 | car theft      |       NULL |
|    3 | 2007-11-03 | burglary       |          3 |
|    4 | 2006-11-03 | burglary       |          3 |
|    5 | 2006-08-04 | vandalism      |       NULL |
|    6 | 2006-02-03 | armed robbery  |       NULL |
+------+------------+----------------+------------+
6 rows in set (0.00 sec)
```

Joining Multiple Tables

- Show a crime with the weather for that day and the suspect in the crime

```
select * from
(crime inner join forecast_data
on my_date = date)
inner join suspect
on suspect.id = crime.suspect_id;
```

te	crime	suspect_id	my_date	description	details	id	idu	given_name	family_name	id
09-06-28	armed robbery	1	2009-06-28	cold	NULL	10	NULL	Rowan	Seymour	

Full Outer Join

- Return all rows from each table, even if there is no match in the corresponding table