IS664 Database Programming
Fall 2022

REGEX

Database Programming

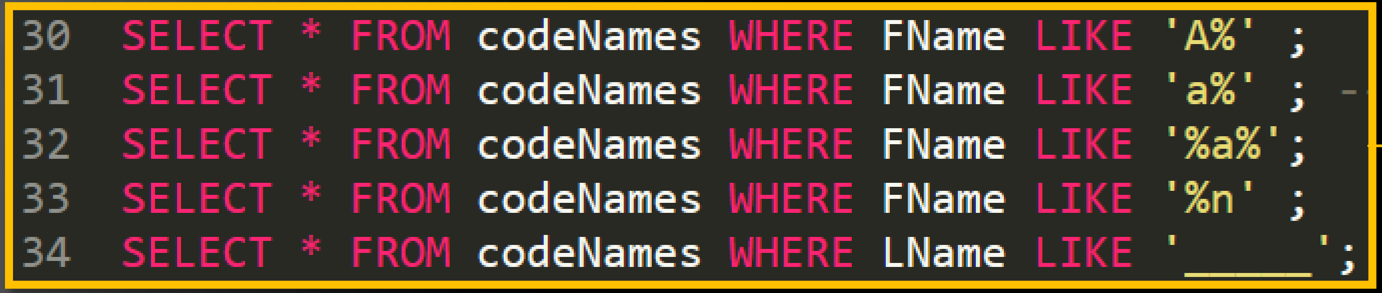
LECTURE 8: REGEX AND TEXT SEARCHING

Professor HG Locklear
hlocklear@pace.edu

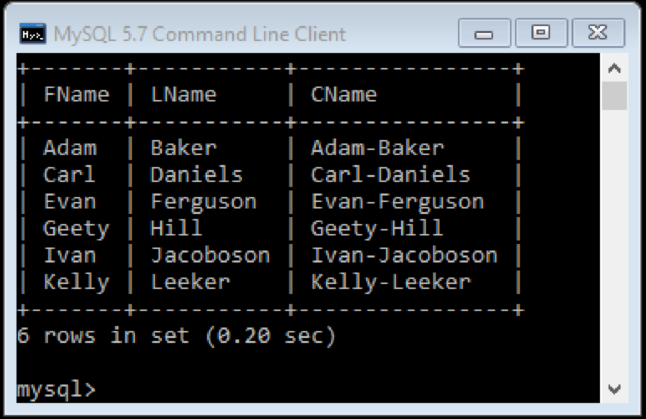
Pattern Matching

- SQL **pattern matching** enables you to use _ to match any single character and % to match an arbitrary number of characters (including zero characters).
- In MySQL, SQL patterns are case-insensitive by default.

- **LIKE 'b%'** anything that begins with **b**
- **LIKE '%fy'** anything that ends with **fy**
- **LIKE '%w%'** anything that contains **w**
- **LIKE '_____'** anything that has five characters

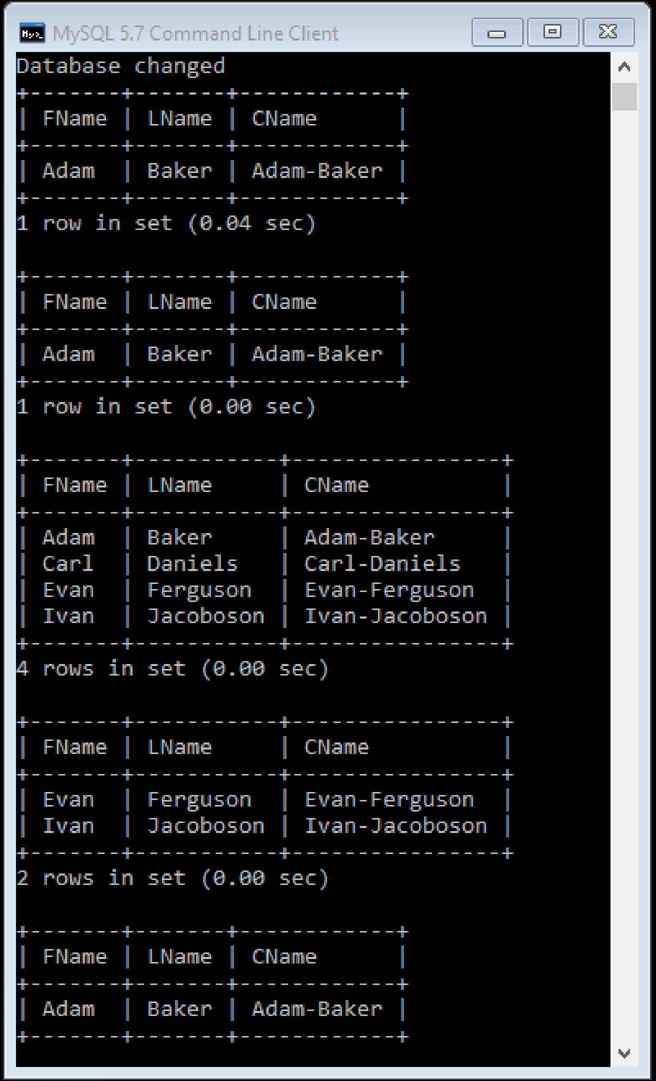


```
SELECT * FROM codeNames WHERE FName LIKE 'A%' ;
SELECT * FROM codeNames WHERE FName LIKE 'a%' ;
SELECT * FROM codeNames WHERE FName LIKE '%a%';
SELECT * FROM codeNames WHERE FName LIKE '%n' ;
SELECT * FROM codeNames WHERE LName LIKE '_____';
```



MySQL 5.7 Command Line Client

```
+-------+-----------+---------------+
| FName | LName     | CName         |
+-------+-----------+---------------+
| Adam  | Baker     | Adam-Baker    |
| Carl  | Daniels   | Carl-Daniels  |
| Evan  | Ferguson  | Evan-Ferguson |
| Geety | Hill      | Geety-Hill    |
| Ivan  | Jacoboson | Ivan-Jacoboson |
| Kelly | Leeker    | Kelly-Leeker  |
+-------+-----------+---------------+
6 rows in set (0.20 sec)

mysql>
```

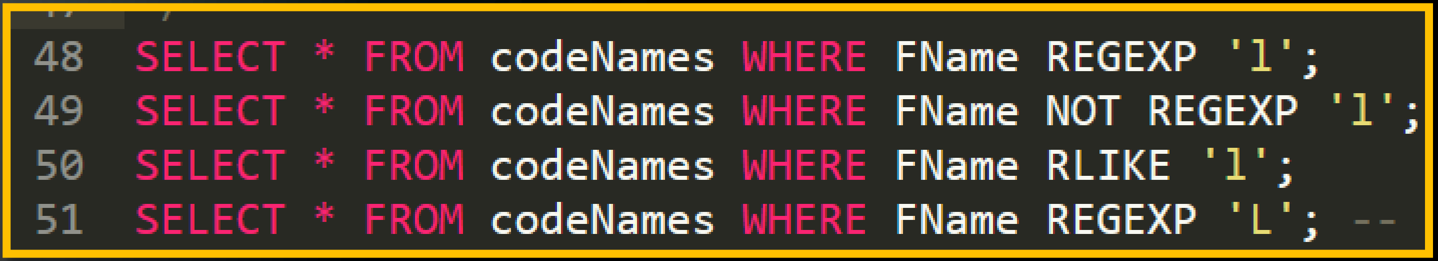


MySQL 5.7 Command Line Client

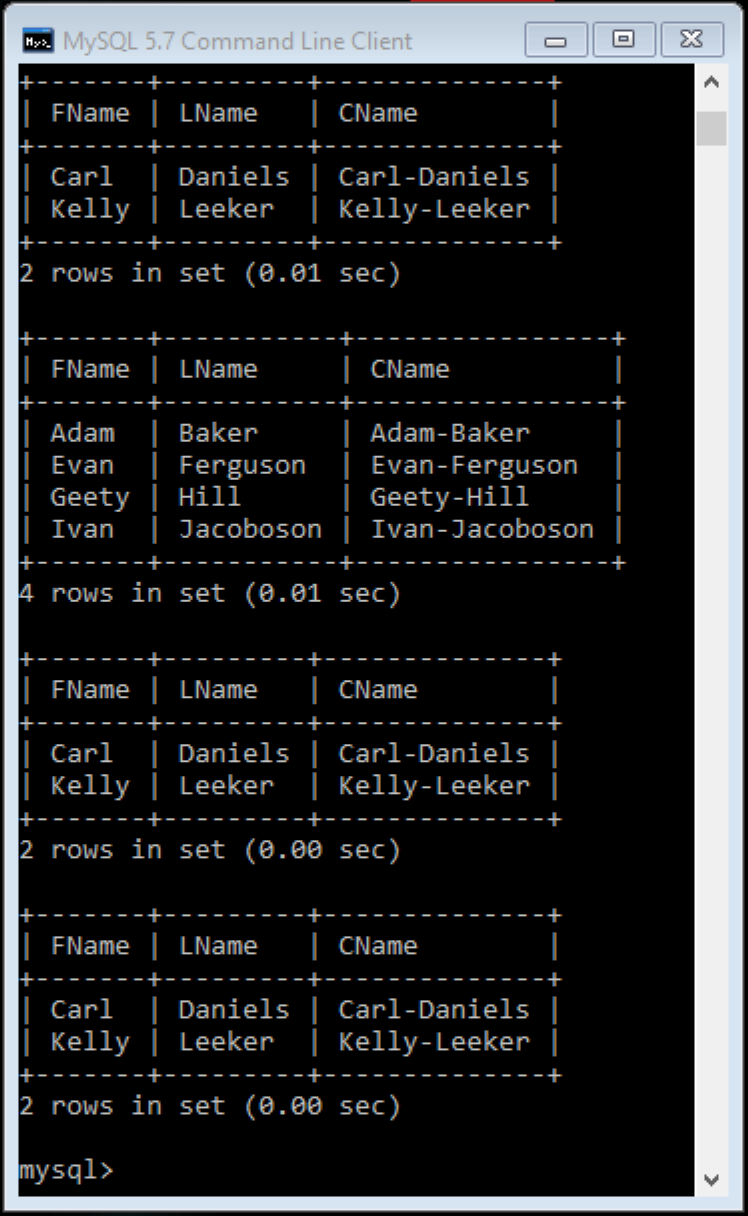
```
Database changed
+-------+-------+------------+
| FName | LName | CName      |
+-------+-------+------------+
| Adam  | Baker | Adam-Baker |
+-------+-------+------------+
1 row in set (0.04 sec)

+-------+-------+------------+
| FName | LName | CName      |
+-------+-------+------------+
| Adam  | Baker | Adam-Baker |
+-------+-------+------------+
1 row in set (0.00 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Adam  | Baker     | Adam-Baker     |
| Carl  | Daniels   | Carl-Daniels   |
| Evan  | Ferguson  | Evan-Ferguson  |
| Ivan  | Jacoboson | Ivan-Jacoboson |
+-------+-----------+----------------+
4 rows in set (0.00 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Evan  | Ferguson  | Evan-Ferguson  |
| Ivan  | Jacoboson | Ivan-Jacoboson |
+-------+-----------+----------------+
2 rows in set (0.00 sec)

+-------+-------+------------+
| FName | LName | CName      |
+-------+-------+------------+
| Adam  | Baker | Adam-Baker |
+-------+-------+------------+
```

REGEX Operators

- A **Regular Expression** is a powerful way of specifying a pattern for a complex search.
- Regular Expression Operators
  - **NOT REGEXP** (Negation of REGEXP)
  - **REGEXP** (True if a String matches the regular expression…false otherwise)
  - **RLIKE** (True if a String matches the regular expression…false otherwise)
- A regular expression pattern match succeeds **if the pattern matches anywhere** in the value being tested.
  - **This differs from a LIKE pattern match, which succeeds only if the pattern matches the entire value.**



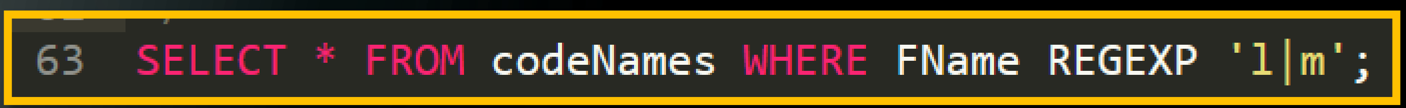
```
SELECT * FROM codeNames WHERE FName REGEXP 'l';
SELECT * FROM codeNames WHERE FName NOT REGEXP 'l';
SELECT * FROM codeNames WHERE FName RLIKE 'l';
SELECT * FROM codeNames WHERE FName REGEXP 'L'; --
```



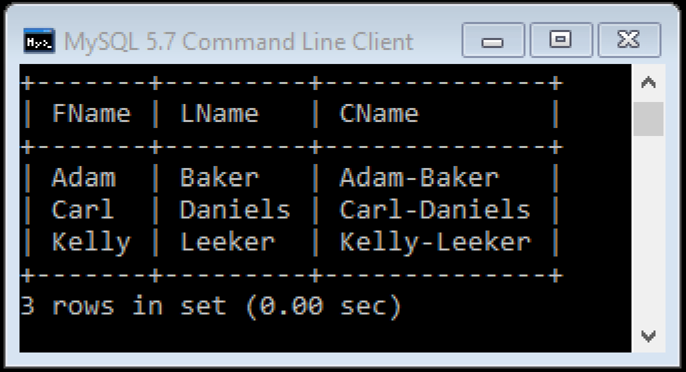
```
+-------+---------+--------------+
| FName | LName   | CName        |
+-------+---------+--------------+
| Carl  | Daniels | Carl-Daniels |
| Kelly | Leeker  | Kelly-Leeker |
+-------+---------+--------------+
2 rows in set (0.01 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Adam  | Baker     | Adam-Baker     |
| Evan  | Ferguson  | Evan-Ferguson  |
| Geety | Hill      | Geety-Hill     |
| Ivan  | Jacoboson | Ivan-Jacoboson |
+-------+-----------+----------------+
4 rows in set (0.01 sec)

+-------+---------+--------------+
| FName | LName   | CName        |
+-------+---------+--------------+
| Carl  | Daniels | Carl-Daniels |
| Kelly | Leeker  | Kelly-Leeker |
+-------+---------+--------------+
2 rows in set (0.00 sec)

+-------+---------+--------------+
| FName | LName   | CName        |
+-------+---------+--------------+
| Carl  | Daniels | Carl-Daniels |
| Kelly | Leeker  | Kelly-Leeker |
+-------+---------+--------------+
2 rows in set (0.00 sec)

mysql>
```

Regular Expressions

- A **Regular Expression** describes a set of strings.
- The simplest regular expression is one that has no special characters in it.
- For example, the regular expression **hello** matches **hello** and nothing else.
- Nontrivial regular expressions use certain special constructs so that they can match more than one string.
- For example, the regular expression **hello|world** contains the **|** alternation operator and matches either the **hello** or **world**.



```
SELECT * FROM codeNames WHERE FName REGEXP 'l|m';
```



```
+-------+---------+--------------+
| FName | LName   | CName        |
+-------+---------+--------------+
| Adam  | Baker   | Adam-Baker   |
| Carl  | Daniels | Carl-Daniels |
| Kelly | Leeker  | Kelly-Leeker |
+-------+---------+--------------+
3 rows in set (0.00 sec)
```

Regular Expressions

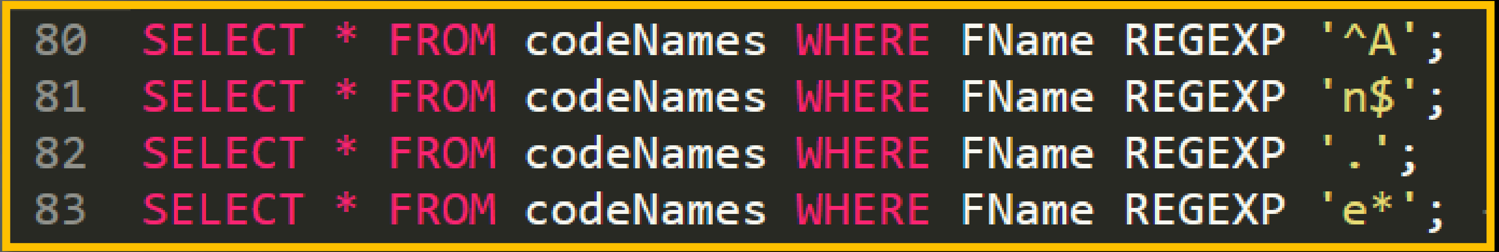
- As a more complex example, the regular expression **B[an]*s** matches any of the strings Bananas, Baaaaas, Bs, and any other string starting with a B, ending with an s, and containing any number of a or n characters in between.
- **^** Match the beginning of a String
- **$** Match the end of a String
- **.** Match any character (including newline)
- ***** Match any sequence of 0 or more characters
- **+** Match any sequence of one or more characters
- **?** Match either 0 or 1 character
- **|** Match either of the sequences
- **()*** Match 0 or more instances of the sequence in the parentheses

Alternative Notation

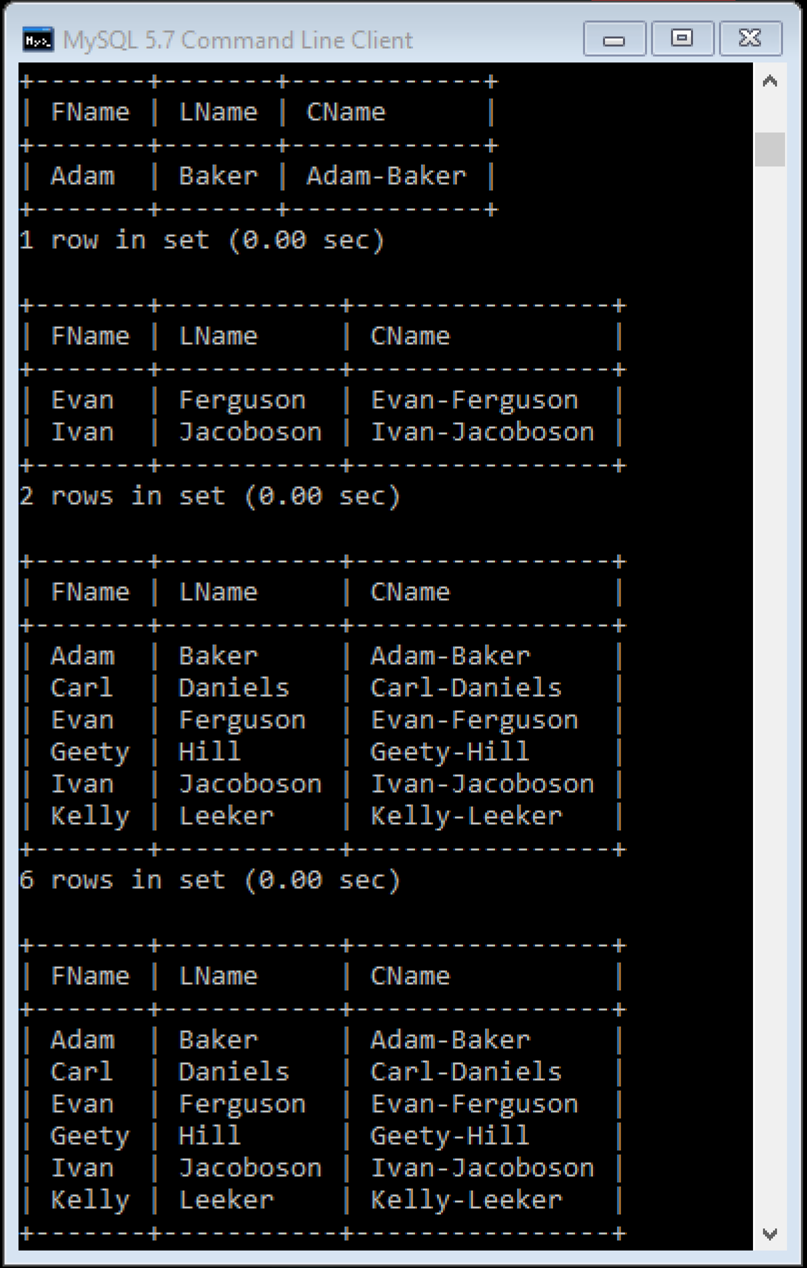
{n},{m,n}

a* could be written as **a{0,}**
a+ could be written as **a{1,}**
a? could be written as **a{0,1}**

Regular Expressions



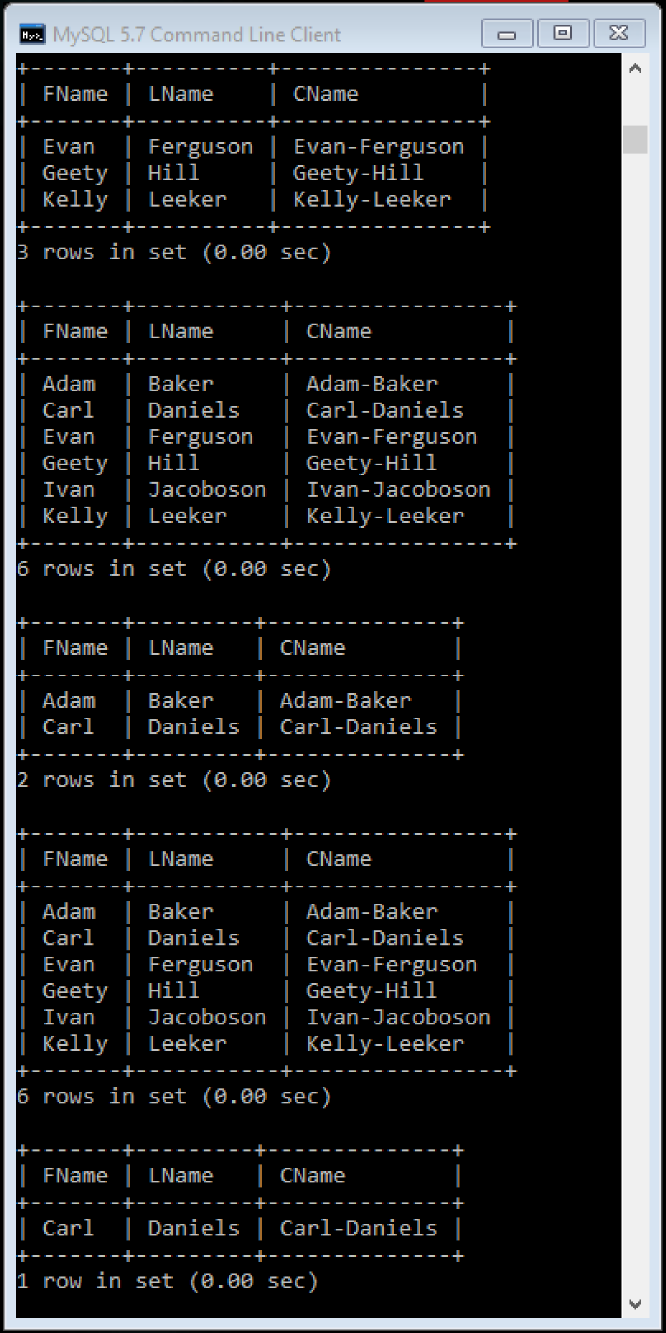
```
SELECT * FROM codeNames WHERE FName REGEXP '^A';
SELECT * FROM codeNames WHERE FName REGEXP 'n$';
SELECT * FROM codeNames WHERE FName REGEXP '.';
SELECT * FROM codeNames WHERE FName REGEXP 'e*';
```



```
+-------+-------+------------+
| FName | LName | CName      |
+-------+-------+------------+
| Adam  | Baker | Adam-Baker |
+-------+-------+------------+
1 row in set (0.00 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Evan  | Ferguson  | Evan-Ferguson  |
| Ivan  | Jacoboson | Ivan-Jacoboson |
+-------+-----------+----------------+
2 rows in set (0.00 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Adam  | Baker     | Adam-Baker     |
| Carl  | Daniels   | Carl-Daniels   |
| Evan  | Ferguson  | Evan-Ferguson  |
| Geety | Hill      | Geety-Hill     |
| Ivan  | Jacoboson | Ivan-Jacoboson |
| Kelly | Leeker    | Kelly-Leeker   |
+-------+-----------+----------------+
6 rows in set (0.00 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Adam  | Baker     | Adam-Baker     |
| Carl  | Daniels   | Carl-Daniels   |
| Evan  | Ferguson  | Evan-Ferguson  |
| Geety | Hill      | Geety-Hill     |
| Ivan  | Jacoboson | Ivan-Jacoboson |
| Kelly | Leeker    | Kelly-Leeker   |
+-------+-----------+----------------+
```

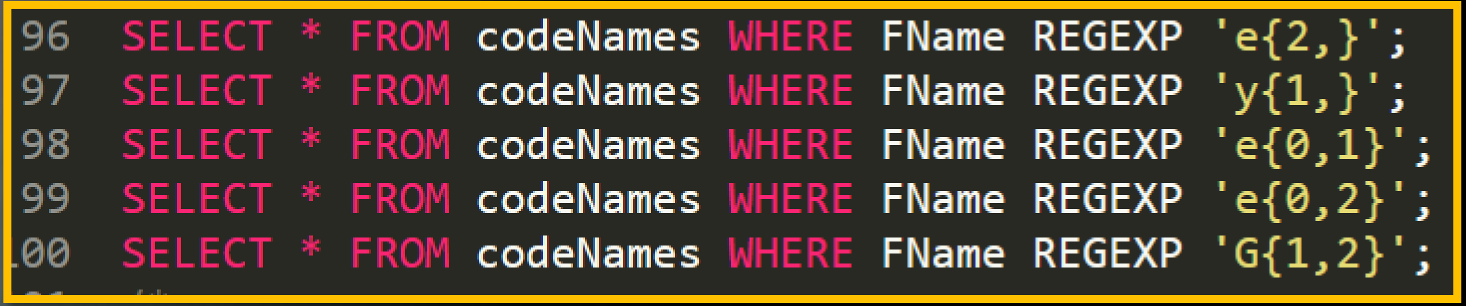
Regular Expressions

```sql
SELECT * FROM codeNames WHERE FName REGEXP 'e+'; -- c
SELECT * FROM codeNames WHERE FName REGEXP 'e?'; -- z
SELECT * FROM codeNames WHERE FName REGEXP 'Ad|rl';
SELECT * FROM codeNames WHERE CName REGEXP '(rl-)*';
SELECT * FROM codeNames WHERE CName REGEXP '(rl-)+';
```

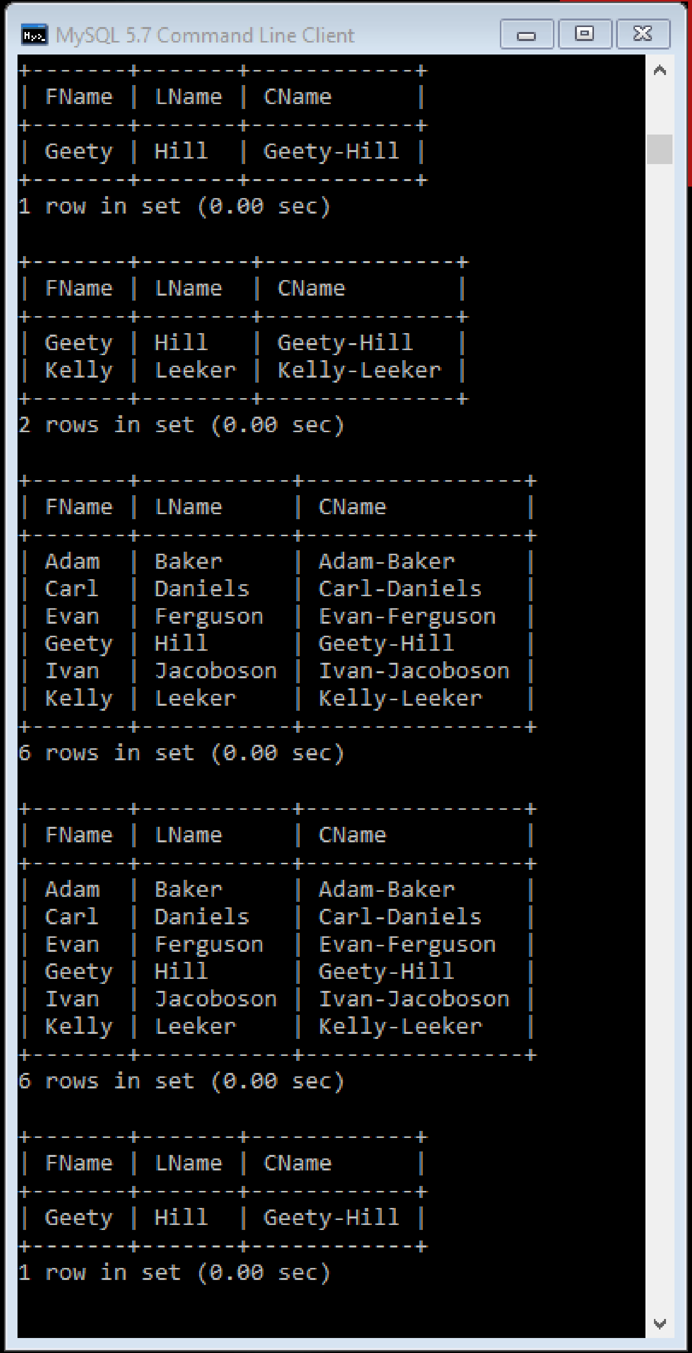

MySQL 5.7 Command Line Client

```
+-------+----------+----------------+
| FName | LName    | CName          |
+-------+----------+----------------+
| Evan  | Ferguson | Evan-Ferguson  |
| Geety | Hill     | Geety-Hill     |
| Kelly | Leeker   | Kelly-Leeker   |
+-------+----------+----------------+
3 rows in set (0.00 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Adam  | Baker     | Adam-Baker     |
| Carl  | Daniels   | Carl-Daniels   |
| Evan  | Ferguson  | Evan-Ferguson  |
| Geety | Hill      | Geety-Hill     |
| Ivan  | Jacoboson | Ivan-Jacoboson |
| Kelly | Leeker    | Kelly-Leeker   |
+-------+-----------+----------------+
6 rows in set (0.00 sec)

+-------+---------+--------------+
| FName | LName   | CName        |
+-------+---------+--------------+
| Adam  | Baker   | Adam-Baker   |
| Carl  | Daniels | Carl-Daniels |
+-------+---------+--------------+
2 rows in set (0.00 sec)

+-------+-----------+----------------+
| FName | LName     | CName          |
+-------+-----------+----------------+
| Adam  | Baker     | Adam-Baker     |
| Carl  | Daniels   | Carl-Daniels   |
| Evan  | Ferguson  | Evan-Ferguson  |
| Geety | Hill      | Geety-Hill     |
| Ivan  | Jacoboson | Ivan-Jacoboson |
| Kelly | Leeker    | Kelly-Leeker   |
+-------+-----------+----------------+
6 rows in set (0.00 sec)

+-------+---------+--------------+
| FName | LName   | CName        |
+-------+---------+--------------+
| Carl  | Daniels | Carl-Daniels |
+-------+---------+--------------+
1 row in set (0.00 sec)
```

Regular Expressions



```
SELECT * FROM codeNames WHERE FName REGEXP 'e{2,}';
SELECT * FROM codeNames WHERE FName REGEXP 'y{1,}';
SELECT * FROM codeNames WHERE FName REGEXP 'e{0,1}';
SELECT * FROM codeNames WHERE FName REGEXP 'e{0,2}';
SELECT * FROM codeNames WHERE FName REGEXP 'G{1,2}';
```

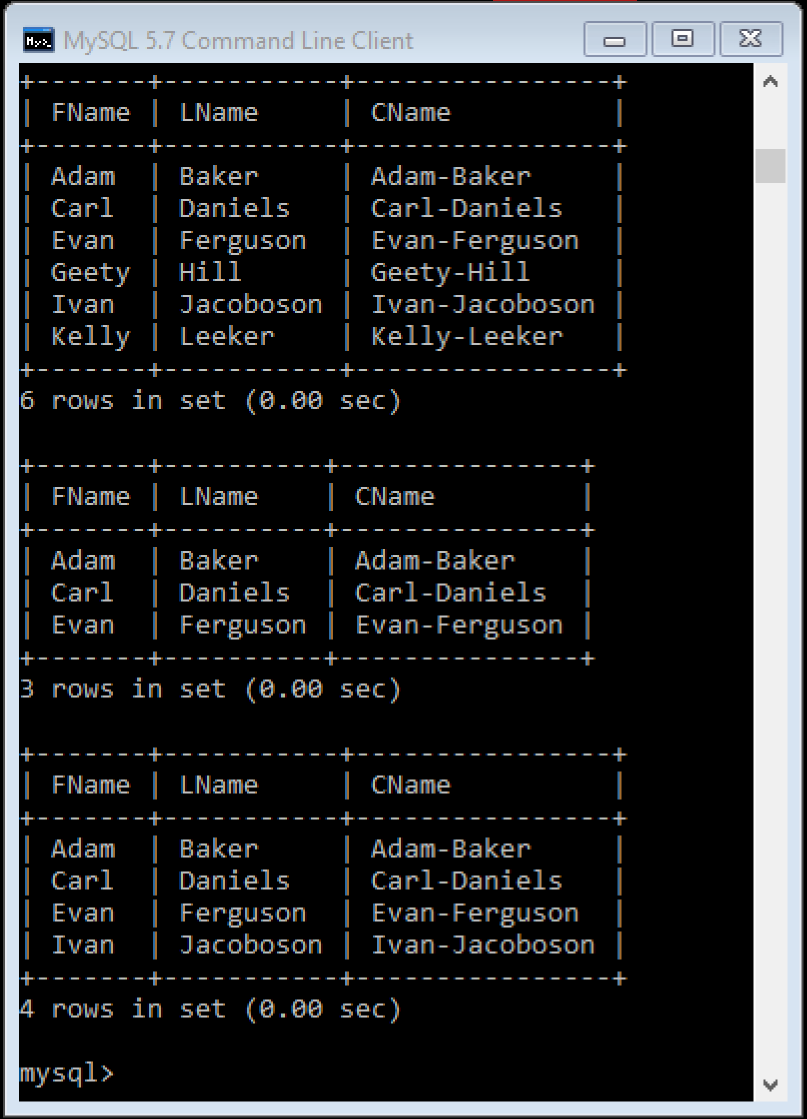


MySQL 5.7 Command Line Client

```
+-------+-------+------------+
| FName | LName | CName      |
+-------+-------+------------+
| Geety | Hill  | Geety-Hill |
+-------+-------+------------+
1 row in set (0.00 sec)

+-------+--------+--------------+
| FName | LName  | CName        |
+-------+--------+--------------+
| Geety | Hill   | Geety-Hill   |
| Kelly | Leeker | Kelly-Leeker |
+-------+--------+--------------+
2 rows in set (0.00 sec)

+-------+-----------+-----------------+
| FName | LName     | CName           |
+-------+-----------+-----------------+
| Adam  | Baker     | Adam-Baker      |
| Carl  | Daniels   | Carl-Daniels    |
| Evan  | Ferguson  | Evan-Ferguson   |
| Geety | Hill      | Geety-Hill      |
| Ivan  | Jacoboson | Ivan-Jacoboson  |
| Kelly | Leeker    | Kelly-Leeker    |
+-------+-----------+-----------------+
6 rows in set (0.00 sec)

+-------+-----------+-----------------+
| FName | LName     | CName           |
+-------+-----------+-----------------+
| Adam  | Baker     | Adam-Baker      |
| Carl  | Daniels   | Carl-Daniels    |
| Evan  | Ferguson  | Evan-Ferguson   |
| Geety | Hill      | Geety-Hill      |
| Ivan  | Jacoboson | Ivan-Jacoboson  |
| Kelly | Leeker    | Kelly-Leeker    |
+-------+-----------+-----------------+
6 rows in set (0.00 sec)

+-------+-------+------------+
| FName | LName | CName      |
+-------+-------+------------+
| Geety | Hill  | Geety-Hill |
+-------+-------+------------+
1 row in set (0.00 sec)
```

Regular Expressions

```
SELECT * FROM codeNames WHERE FName REGEXP '^[A-K]';
SELECT * FROM codeNames WHERE FName REGEXP '^[a-e]';
SELECT * FROM codeNames WHERE FName REGEXP '[l-n]$';
```



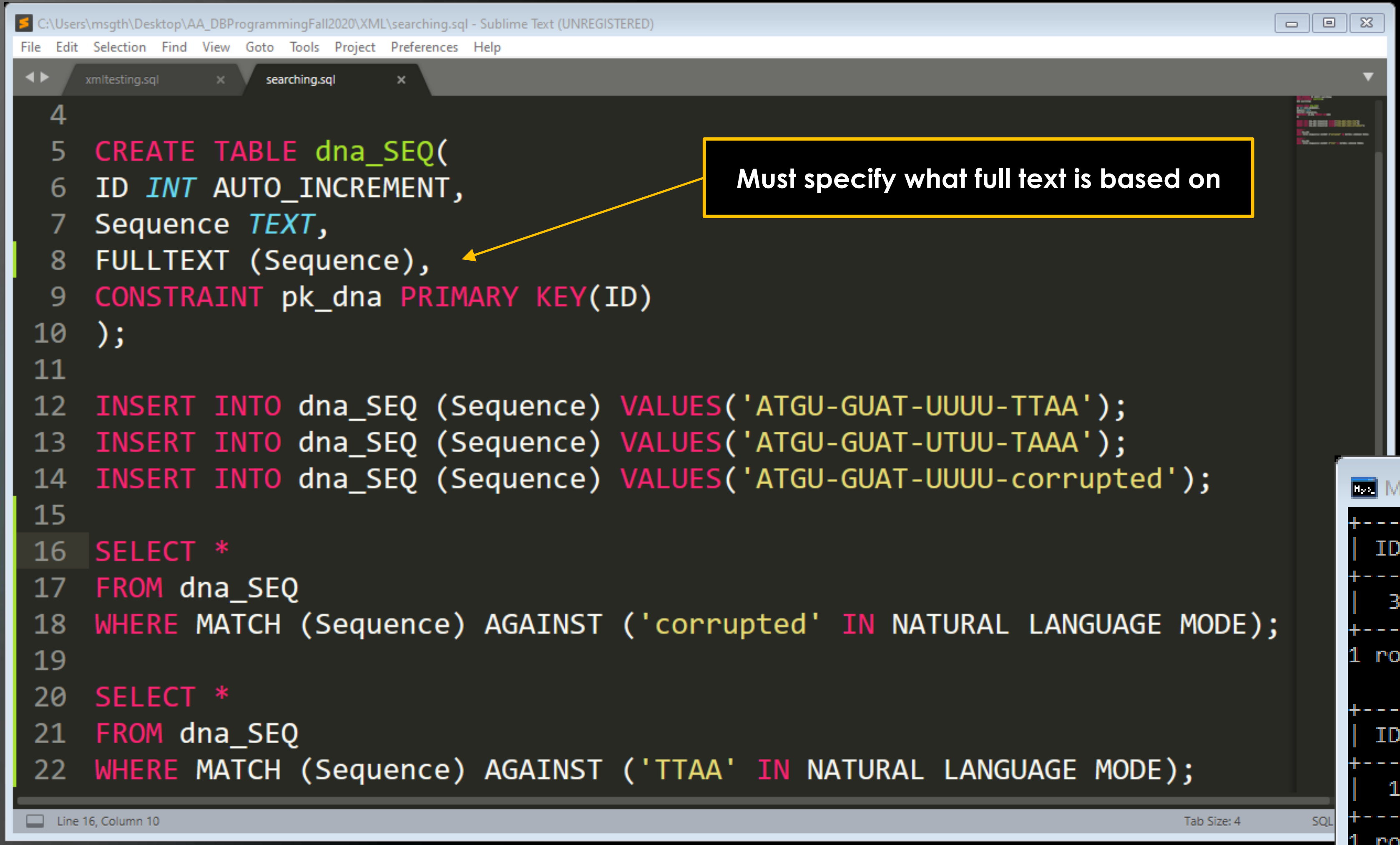
MySQL 5.7 Command Line Client

```
+-------+-----------+-----------------+
| FName | LName     | CName           |
+-------+-----------+-----------------+
| Adam  | Baker     | Adam-Baker      |
| Carl  | Daniels   | Carl-Daniels    |
| Evan  | Ferguson  | Evan-Ferguson   |
| Geety | Hill      | Geety-Hill      |
| Ivan  | Jacoboson | Ivan-Jacoboson  |
| Kelly | Leeker    | Kelly-Leeker    |
+-------+-----------+-----------------+
6 rows in set (0.00 sec)

+-------+----------+---------------+
| FName | LName    | CName         |
+-------+----------+---------------+
| Adam  | Baker    | Adam-Baker    |
| Carl  | Daniels  | Carl-Daniels  |
| Evan  | Ferguson | Evan-Ferguson |
+-------+----------+---------------+
3 rows in set (0.00 sec)

+-------+-----------+-----------------+
| FName | LName     | CName           |
+-------+-----------+-----------------+
| Adam  | Baker     | Adam-Baker      |
| Carl  | Daniels   | Carl-Daniels    |
| Evan  | Ferguson  | Evan-Ferguson   |
| Ivan  | Jacoboson | Ivan-Jacoboson  |
+-------+-----------+-----------------+
4 rows in set (0.00 sec)

mysql>
```

Full Text Searching

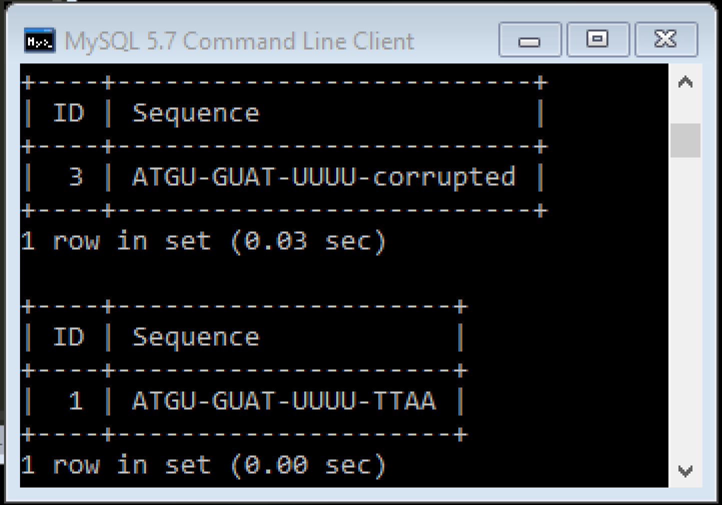
- In MySQL we have the concept of **full-text** searching.
- Full-Text searching is utilizing the MATCH function and the AGAINST operator to takes a comma-separated list that specifies the attributes to be searched and a string to search for and an optional modifier that indicates what type of search to perform.
- There are three types of full-text searches:
  - **Natural Language Search**
    - Interprets the search string as a phrase in natural human language.
    - There are no special operators, with the exception of double quote (") characters.
  - **Boolean Search**
    - A boolean search interprets the search string using the rules of a special query language.
    - It can also contain operators that specify requirements for the words being searched for.
  - ***Query Expansion Search (we will not explore this in our course)***
    - Is a modification of a natural language search.
    - The words from the most relevant rows returned by the search are added to the search string and the search is done again.
    - The query returns the rows from the second search.

Full Text Searching


C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\XML\searching.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
xmltesting.sql
searching.sql
Line 16, Column 10
Tab Size: 4
SQL

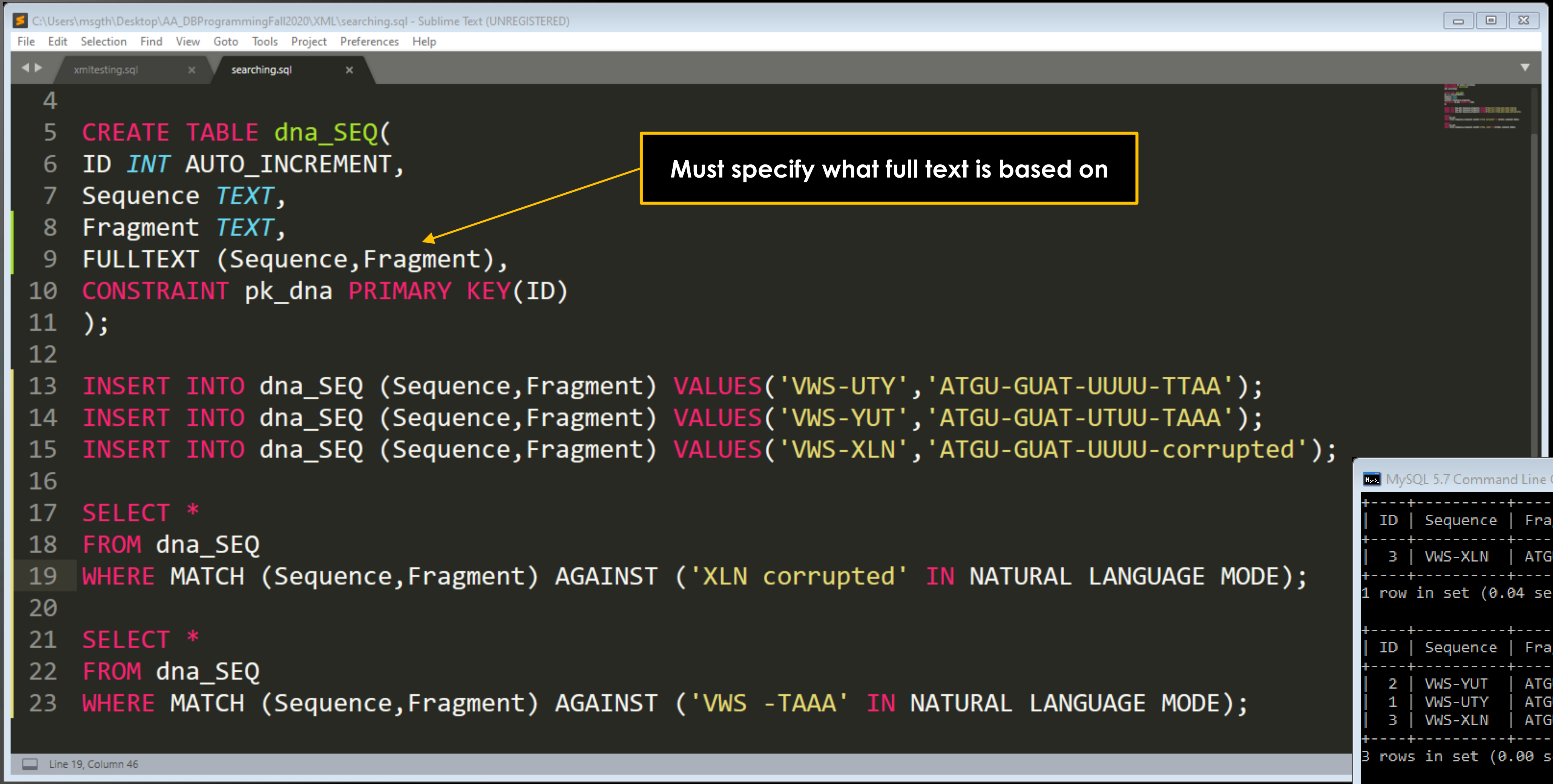
```sql

CREATE TABLE dna_SEQ(
ID INT AUTO_INCREMENT,
Sequence TEXT,
FULLTEXT (Sequence),
CONSTRAINT pk_dna PRIMARY KEY(ID)
);

INSERT INTO dna_SEQ (Sequence) VALUES('ATGU-GUAT-UUUU-TTAA');
INSERT INTO dna_SEQ (Sequence) VALUES('ATGU-GUAT-UTUU-TAAA');
INSERT INTO dna_SEQ (Sequence) VALUES('ATGU-GUAT-UUUU-corrupted');

SELECT *
FROM dna_SEQ
WHERE MATCH (Sequence) AGAINST ('corrupted' IN NATURAL LANGUAGE MODE);

SELECT *
FROM dna_SEQ
WHERE MATCH (Sequence) AGAINST ('TTAA' IN NATURAL LANGUAGE MODE);
```

Must specify what full text is based on



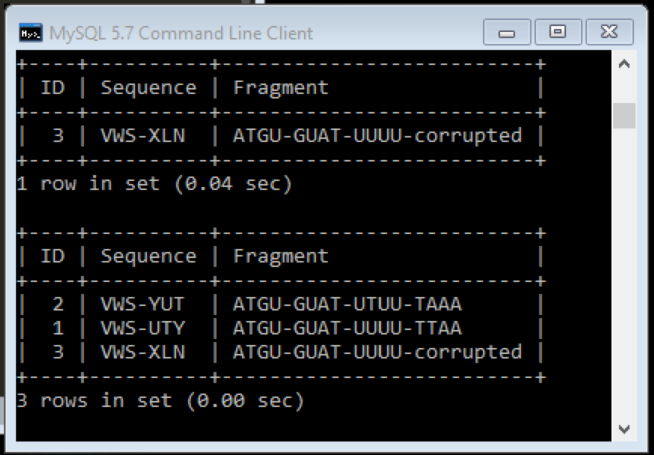
```
+----+--------------------------+
| ID | Sequence                 |
+----+--------------------------+
|  3 | ATGU-GUAT-UUUU-corrupted |
+----+--------------------------+
1 row in set (0.03 sec)

+----+---------------------+
| ID | Sequence            |
+----+---------------------+
|  1 | ATGU-GUAT-UUUU-TTAA |
+----+---------------------+
1 row in set (0.00 sec)
```

Natural Language


C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\XML\searching.sql - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
xmltesting.sql
searching.sql
Line 19, Column 46

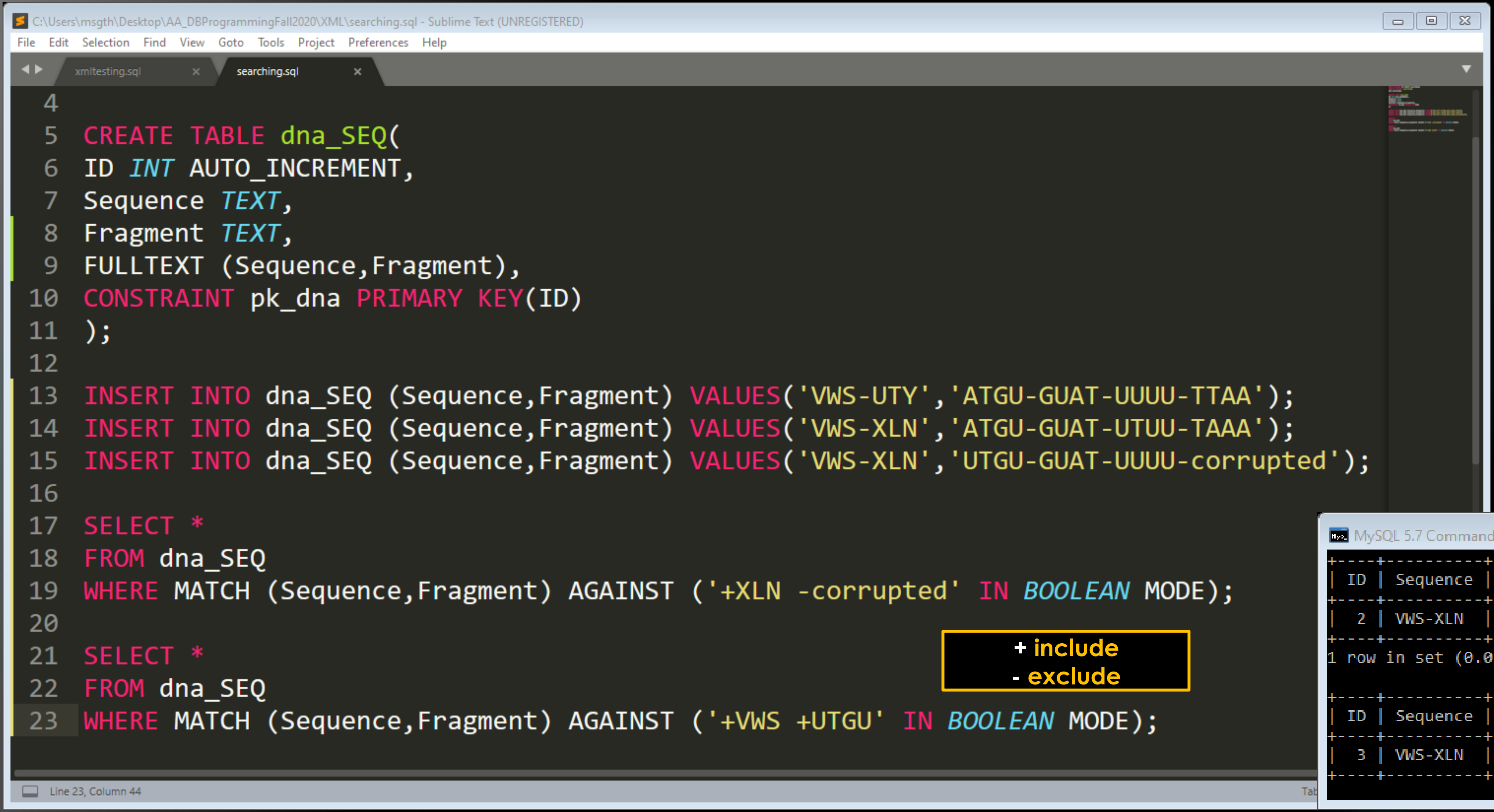
```sql

CREATE TABLE dna_SEQ(
ID INT AUTO_INCREMENT,
Sequence TEXT,
Fragment TEXT,
FULLTEXT (Sequence,Fragment),
CONSTRAINT pk_dna PRIMARY KEY(ID)
);

INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-UTY','ATGU-GUAT-UUUU-TTAA');
INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-YUT','ATGU-GUAT-UTUU-TAAA');
INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-XLN','ATGU-GUAT-UUUU-corrupted');

SELECT *
FROM dna_SEQ
WHERE MATCH (Sequence,Fragment) AGAINST ('XLN corrupted' IN NATURAL LANGUAGE MODE);

SELECT *
FROM dna_SEQ
WHERE MATCH (Sequence,Fragment) AGAINST ('VWS -TAAA' IN NATURAL LANGUAGE MODE);
```

Must specify what full text is based on



```
+----+----------+--------------------------+
| ID | Sequence | Fragment                 |
+----+----------+--------------------------+
|  3 | VWS-XLN  | ATGU-GUAT-UUUU-corrupted |
+----+----------+--------------------------+
1 row in set (0.04 sec)

+----+----------+--------------------------+
| ID | Sequence | Fragment                 |
+----+----------+--------------------------+
|  2 | VWS-YUT  | ATGU-GUAT-UTUU-TAAA      |
|  1 | VWS-UTY  | ATGU-GUAT-UUUU-TTAA      |
|  3 | VWS-XLN  | ATGU-GUAT-UUUU-corrupted |
+----+----------+--------------------------+
3 rows in set (0.00 sec)
```

Boolean

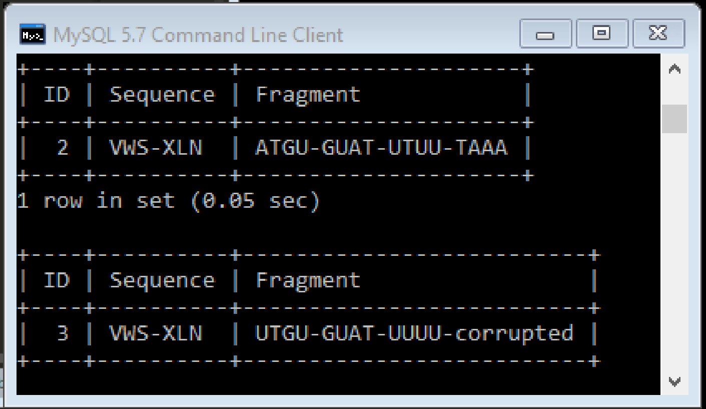


C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\XML\searching.sql - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help
xmltesting.sql
searching.sql

```sql

CREATE TABLE dna_SEQ(
ID INT AUTO_INCREMENT,
Sequence TEXT,
Fragment TEXT,
FULLTEXT (Sequence,Fragment),
CONSTRAINT pk_dna PRIMARY KEY(ID)
);

INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-UTY','ATGU-GUAT-UUUU-TTAA');
INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-XLN','ATGU-GUAT-UTUU-TAAA');
INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-XLN','UTGU-GUAT-UUUU-corrupted');

SELECT *
FROM dna_SEQ
WHERE MATCH (Sequence,Fragment) AGAINST ('+XLN -corrupted' IN BOOLEAN MODE);

SELECT *
FROM dna_SEQ
WHERE MATCH (Sequence,Fragment) AGAINST ('+VWS +UTGU' IN BOOLEAN MODE);
```

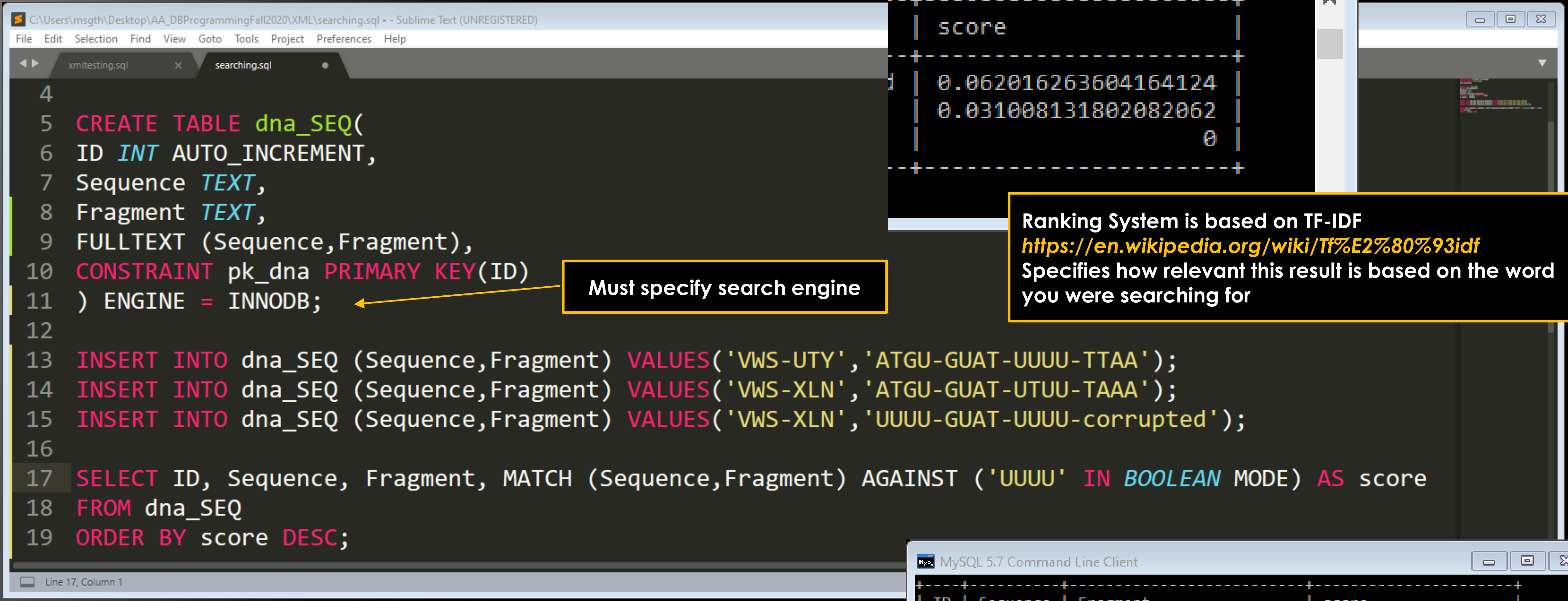
+ include
- exclude



MySQL 5.7 Command Line Client

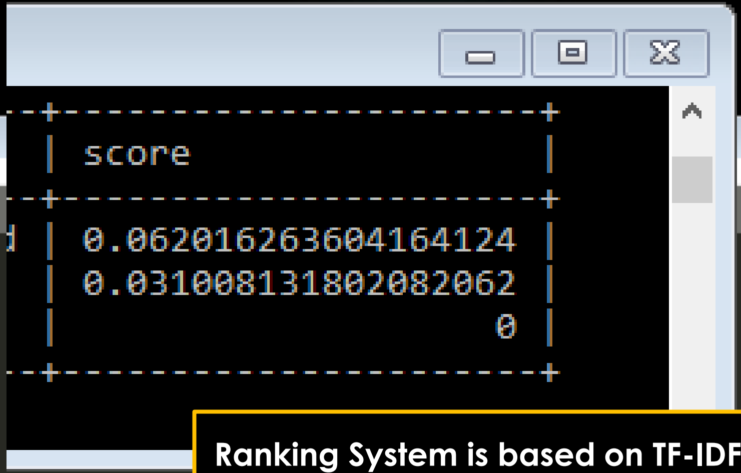
```
+----+----------+---------------------+
| ID | Sequence | Fragment            |
+----+----------+---------------------+
|  2 | VWS-XLN  | ATGU-GUAT-UTUU-TAAA |
+----+----------+---------------------+
1 row in set (0.05 sec)

+----+----------+--------------------------+
| ID | Sequence | Fragment                 |
+----+----------+--------------------------+
|  3 | VWS-XLN  | UTGU-GUAT-UUUU-corrupted |
+----+----------+--------------------------+
```

Scoring


C:\Users\msgth\Desktop\AA_DBProgrammingFall2020\XML\searching.sql • - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
xmltesting.sql
searching.sql
Line 17, Column 1

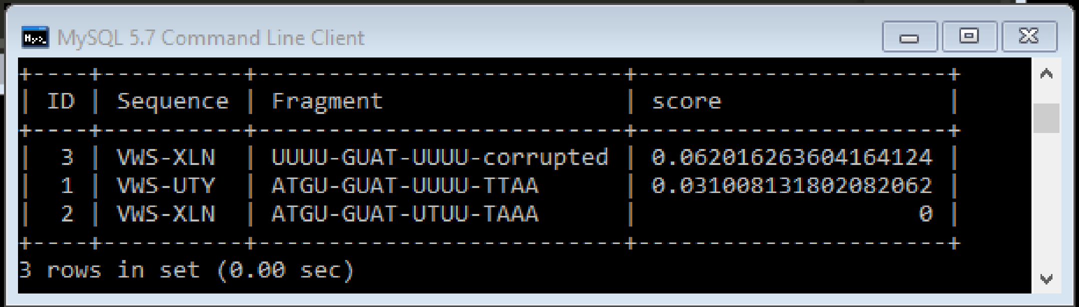
```sql
CREATE TABLE dna_SEQ(
ID INT AUTO_INCREMENT,
Sequence TEXT,
Fragment TEXT,
FULLTEXT (Sequence,Fragment),
CONSTRAINT pk_dna PRIMARY KEY(ID)
) ENGINE = INNODB;

INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-UTY','ATGU-GUAT-UUUU-TTAA');
INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-XLN','ATGU-GUAT-UTUU-TAAA');
INSERT INTO dna_SEQ (Sequence,Fragment) VALUES('VWS-XLN','UUUU-GUAT-UUUU-corrupted');

SELECT ID, Sequence, Fragment, MATCH (Sequence,Fragment) AGAINST ('UUUU' IN BOOLEAN MODE) AS score
FROM dna_SEQ
ORDER BY score DESC;
```

Must specify search engine



```
-+-----------------------+
 | score                 |
-+-----------------------+
d | 0.062016263604164124 |
  | 0.031008131802082062 |
  |                    0 |
-+-----------------------+
```

Ranking System is based on TF-IDF
https://en.wikipedia.org/wiki/Tf%E2%80%93idf
Specifies how relevant this result is based on the word you were searching for



```
+----+----------+--------------------------+----------------------+
| ID | Sequence | Fragment                 | score                |
+----+----------+--------------------------+----------------------+
|  3 | VWS-XLN  | UUUU-GUAT-UUUU-corrupted | 0.062016263604164124 |
|  1 | VWS-UTY  | ATGU-GUAT-UUUU-TTAA      | 0.031008131802082062 |
|  2 | VWS-XLN  | ATGU-GUAT-UTUU-TAAA      |                    0 |
+----+----------+--------------------------+----------------------+
3 rows in set (0.00 sec)
```