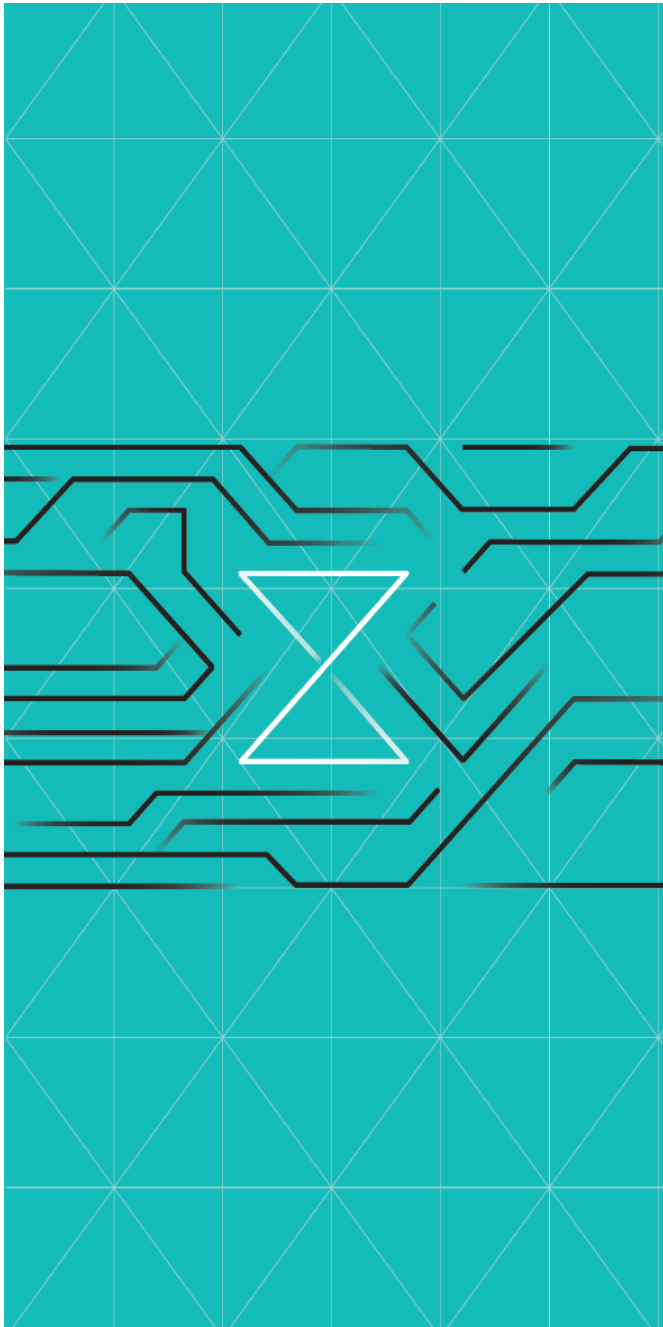


SQL1

SQL and Relational Databases

- [MANAGING DATA AS SETS - RELATIONAL DATABASES](#)
- [1 INSTALL DB2 VSCODE EXTENSION](#)
- [2 INSTALL/CONFIGURE JAVA](#)
- [3 SET UP A LITTLE MORE INFORMATION](#)
- [4 MAKE A STATEMENT](#)
- [5 RESULT!](#)
- [6 GO EVEN DEEPER](#)
- [7 WHERE YOU WANT TO BE](#)
- [8 A SPECIAL SET OF RESULTS](#)
- [9 MAKE IT COUNT \(FOR REAL\)](#)
- [10 ALL ABOUT JOINS](#)
- [11 JOINS HAPPEN](#)
- [12 PURE FILTERED RESULTS](#)
- [13 PUSH IT TO THE MAX](#)
- [14 PUTTING IT ALL TOGETHER](#)
- [15 LAST CHECK AND WRAP-UP](#)



MANAGING DATA AS SETS - RELATIONAL DATABASES

The Challenge

Relational databases are an important part of just about all software today.

By organizing data into tables and then linking these tables based on common data, you can retrieve and organize data in order to better understand relationships in the data.

In this challenge, you will be using SQL to communicate with IBM Db2 relational databases, hosted on the IBM zOS system.

Before You Begin

Not much, just make sure everything's right with VSCode and you should be good to go.

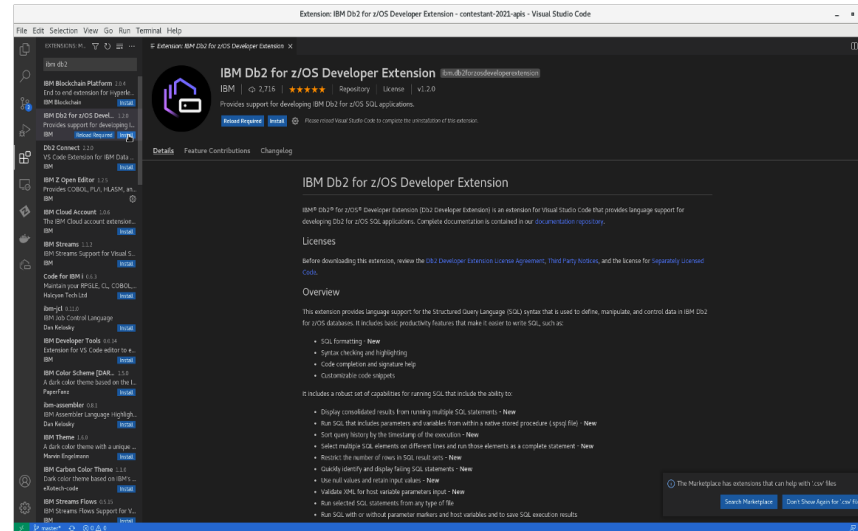
Investment

Steps	Duration
15	60 minutes

SQL1250211-2139

1 INSTALL DB2 VSCODE EXTENSION

In VSCode, on the “Extensions” launcher in the left-side menu, and search for IBM Db2.



Next, click on the “Install” button for “IBM Db2 for z/OS Developer Extension.” This is an extension for VSCode that gives you the ability to work interactively with Db2 databases.

The install should only take a few seconds on most broadband internet connections.

SQL1250211-2139

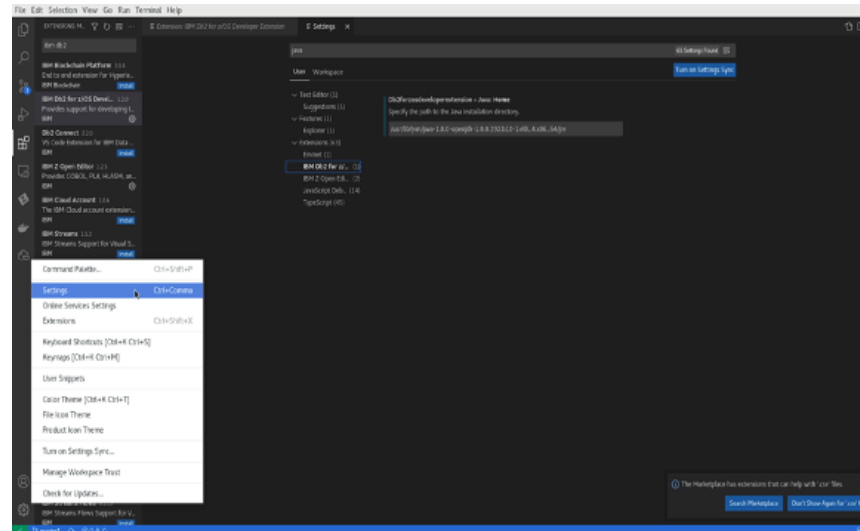
2 INSTALL/CONFIGURE JAVA

For the Db2 extension to be able to connect to the database server, you need to have a supported Java JDK installed.

Not just a Java Runtime (JRE), but the full Java Developer Kit (JDK).

You have options: you can use [Oracle Java JDK 8](#) , or version 8 or 11 of the [OpenJDK](#) .

Once installed, navigate into the VSCode settings and search for “java” in the search bar. There should be an entry for Db2 under Extensions that you can click on and provide the path for your installed Java runtime libraries.



For example, on MacOS it is usually `/Library/Java/JavaVirtualMachine/jdk1.8.0_181.jdk/Contents/Home`

On Windows, it will probably be `C:\Files.0_181`

3 SET UP A LITTLE MORE INFORMATION

In the VSCode left-side menu, navigate to “Db2 Developer Extension” (you may need to click on the 3 dots “. . .” to see it) and click the plus sign to add a new connection profile.

The screenshot shows the 'Add Db2 Connection' dialog in the VS Code Db2 Developer Extension. The left sidebar shows the 'DB2 DEVELOPER EXTENSION' menu with 'DB2 FOR Z/OS CONNECTIONS' expanded. The main area is titled 'Db2 connection information' and contains the following fields:

- Location name: ZXPDB2
- Connection name: 204.90.115.200/ZXPDB2
- Host: 204.90.115.200
- Port: 5040
- Connection URL: jdbc:db2://204.90.115.200:5040/ZXPDB2

Under the 'User information' section:

- Login method: Password (selected)
- User ID: Z#####
- Password: (masked with dots)
- Save password: ☒

A 'Finish' button is located at the bottom right of the dialog.

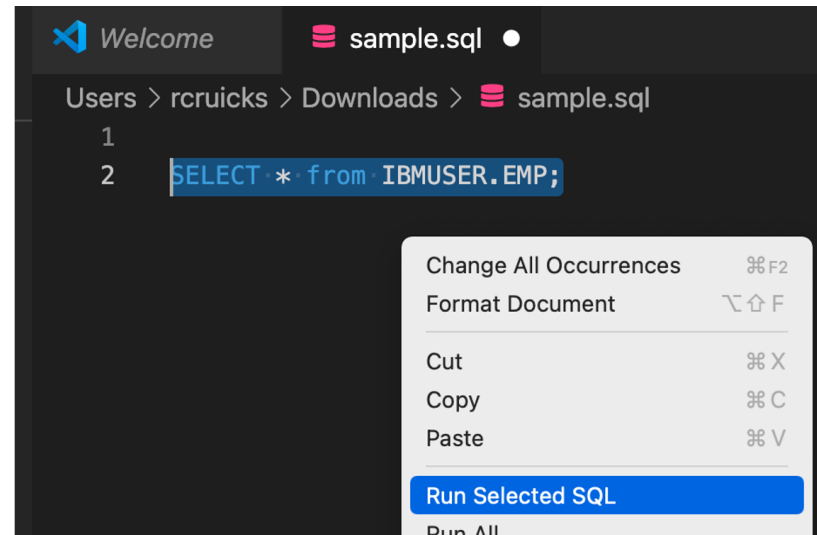
Enter the following values for the profile configuration:

- **ZXPDB2** for location name
- **204.90.115.200** for host
- **5040** for port
- and your z/OS login for User ID and password

Unless you want to type your password every time you start doing SQL queries, check the “Save password” box.

Then click Finish at the bottom and you should be setup and ready to use Db2!

4 MAKE A STATEMENT



Create a new folder on your laptop named “Db2 Challenges”, open this folder in VSCode, and create a new file named *sample.sql* as shown above.

Now write a simple SQL statement. Type in the statement from the screenshot above:

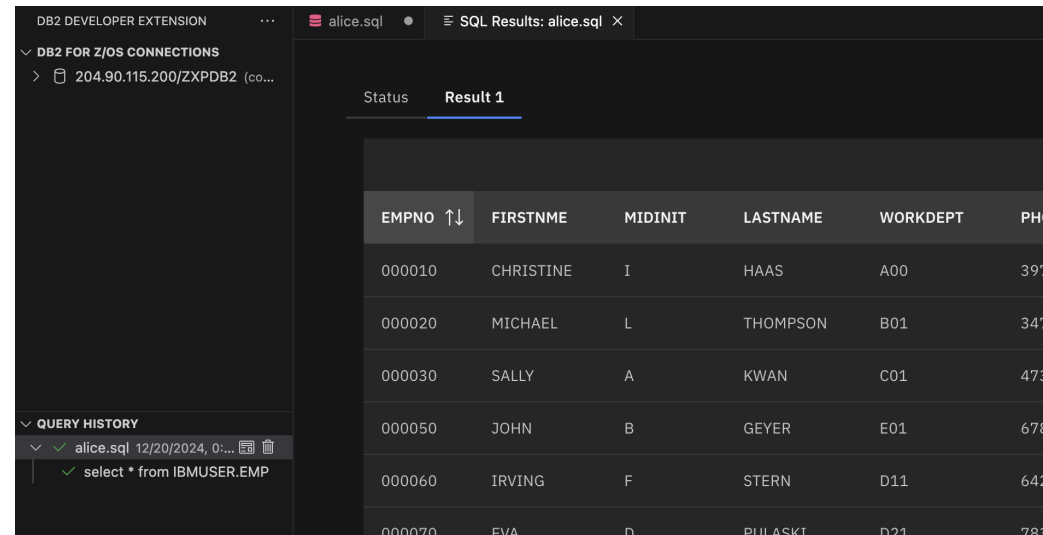
● **SELECT** * **from** IBMUSER.EMP;

Highlight the statement and right-click to see options to run.

Choose “Run Selected SQL”.

SQL1250211-2139

5 RESULT!



The screenshot shows the Db2 Developer Extension interface in VSCode. The left sidebar contains a tree view with 'DB2 FOR Z/OS CONNECTIONS' and a 'QUERY HISTORY' section. The main editor area displays the 'SQL Results: alice.sql' tab, which is split into 'Status' and 'Result 1' sections. The 'Result 1' section shows a table with 7 columns: EMPNO, FIRSTNAME, MIDINIT, LASTNAME, WORKDEPT, and PHO. The table contains 7 rows of employee data.

EMPNO	FIRSTNAME	MIDINIT	LASTNAME	WORKDEPT	PHO
000010	CHRISTINE	I	HAAS	A00	397
000020	MICHAEL	L	THOMPSON	B01	347
000030	SALLY	A	KWAN	C01	473
000050	JOHN	B	GEYER	E01	678
000060	IRVING	F	STERN	D11	642
000070	EVA	D	PULASKI	D21	783

The “SQL Results” tab will pop up after the statement is run and this is where you can view the set of records as a result of your SQL statement.

The “Status” section will display the statement you ran, and Return Code 0 indicating successful execution. The “Result 1” tab contains the resulting table.

As you can see, there are a *lot* of columns in the table. When you put the asterisk (*) after SELECT in your SQL statement, this tells the database to retrieve all the columns from the table *ibmuser.emp*, and include them in the results.

Note the “QUERY HISTORY” section in the left-side navigation area of the Db2 Developer Extension in VSCode.

SQL1250211-2139

6 GO EVEN DEEPER

You may have guessed that if you don't want to work with all the columns, you can specify which columns you do want.

For example, you could display just the employees' job by typing `select JOB from ibmuser.emp`

You can also display multiple columns. Type in

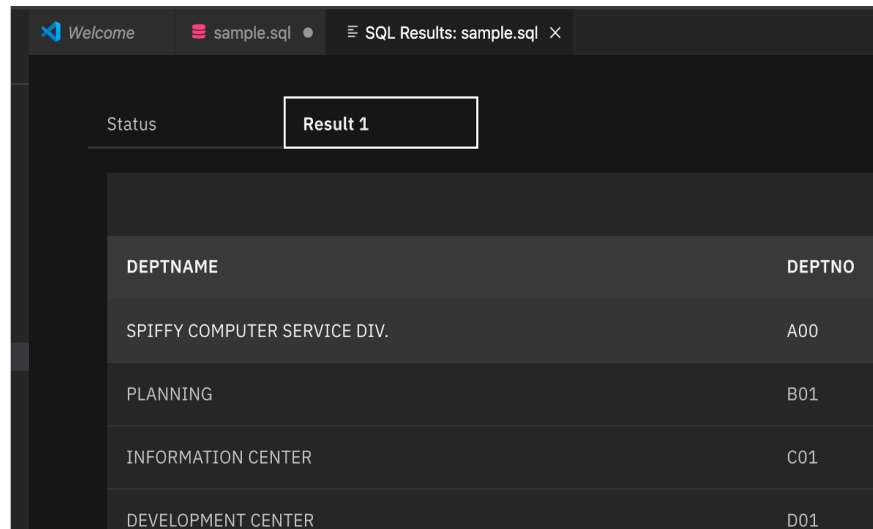
`select LASTNAME, EMPNO, PHONENO from ibmuser.emp`

and see the result.

Note that the columns shows up in the exact order you specified.

Notice the table you've been using is called "ibmuser.emp" and contains some totally fictional employee data.

Well, this is another table you can use called "ibmuser.dept" that contains data about the different departments within this fictional company.



DEPTNAME	DEPTNO
SPIFFY COMPUTER SERVICE DIV.	A00
PLANNING	B01
INFORMATION CENTER	C01
DEVELOPMENT CENTER	D01

Look at the contents of the "ibmuser.dept" table and then write an SQL statement that will recreate the results shown in the screenshot above.

“So what exactly is Db2?”

IBM Db2 is what is known as a Relational Database Management System (RDBMS).

It uses the the concepts of relational databases to store many different types of data and then optimizes retrieval of that data.

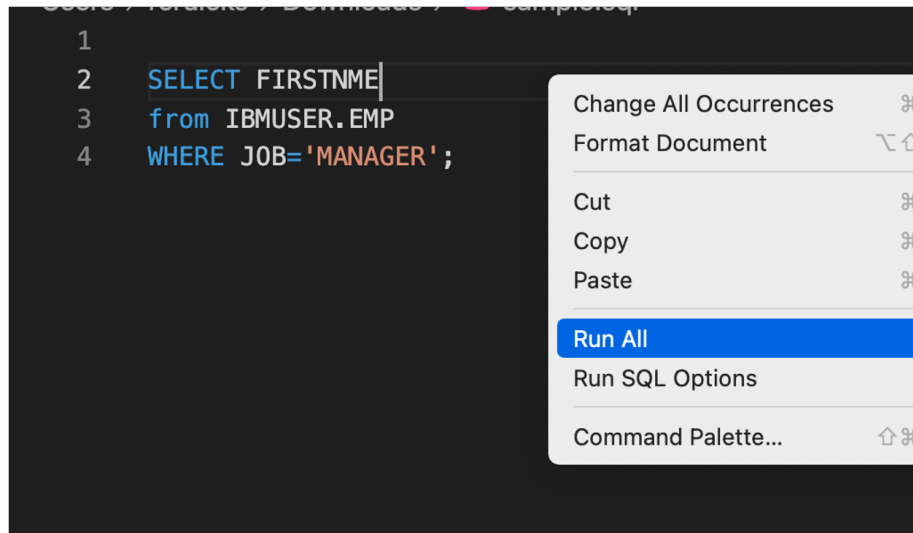
There are many different types of RDBMS, Db2 is just IBM’s version - it has been around since the beginning of relational databases. Actually the concept of a relational database was developed at IBM by computer scientist Edgar Frank “Ted” Codd.

SQL (Structured Query Language) is the language we use to talk to Db2, and is what pretty much all modern RDBMSs use.

7 WHERE YOU WANT TO BE

Not only can you specify the columns you want to select, but you can specify the rows as well.

This is done by specifying the value of one (or more) of the columns so that only rows that meet the column values are displayed.



The screenshot shows a code editor with a dark background. The SQL query is as follows:

```
1  
2 SELECT FIRSTNME  
3 from IBMUSER.EMP  
4 WHERE JOB='MANAGER';
```

A context menu is open on the right side of the editor, with the following options:

- Change All Occurrences
- Format Document
- Cut
- Copy
- Paste
- Run All** (highlighted in blue)
- Run SQL Options
- Command Palette...

Can you guess what the above query will return?

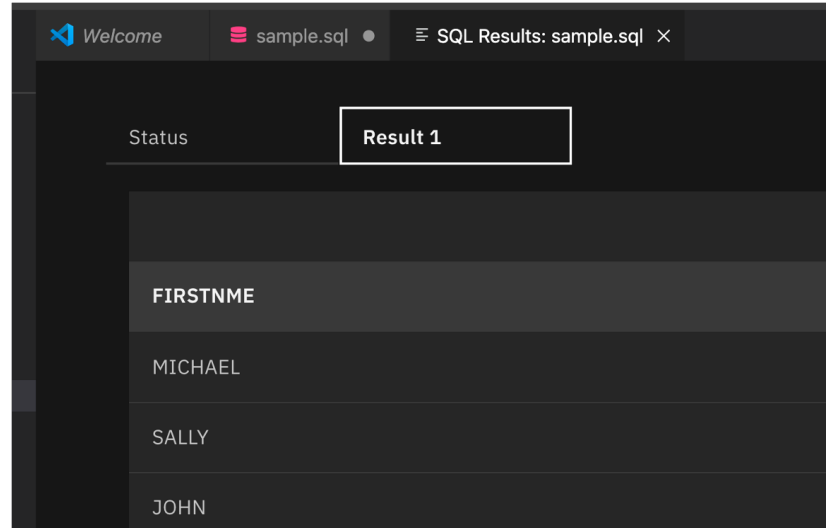
Run the multi-line statement in the screenshot above using the WHERE clause and find out.

- **SELECT** FIRSTNME
- **from** IBMUSER.EMP
- **WHERE** JOB='MANAGER' ;

What do you think would happen for *JOB='manager'*?

8 A SPECIAL SET OF RESULTS

Look at the result from the query, and see what you have.



Status		Result 1
		FIRSTNAME
		MICHAEL
		SALLY
		JOHN

Notice that only those employees that are managers show up. Is that what you expected to happen?

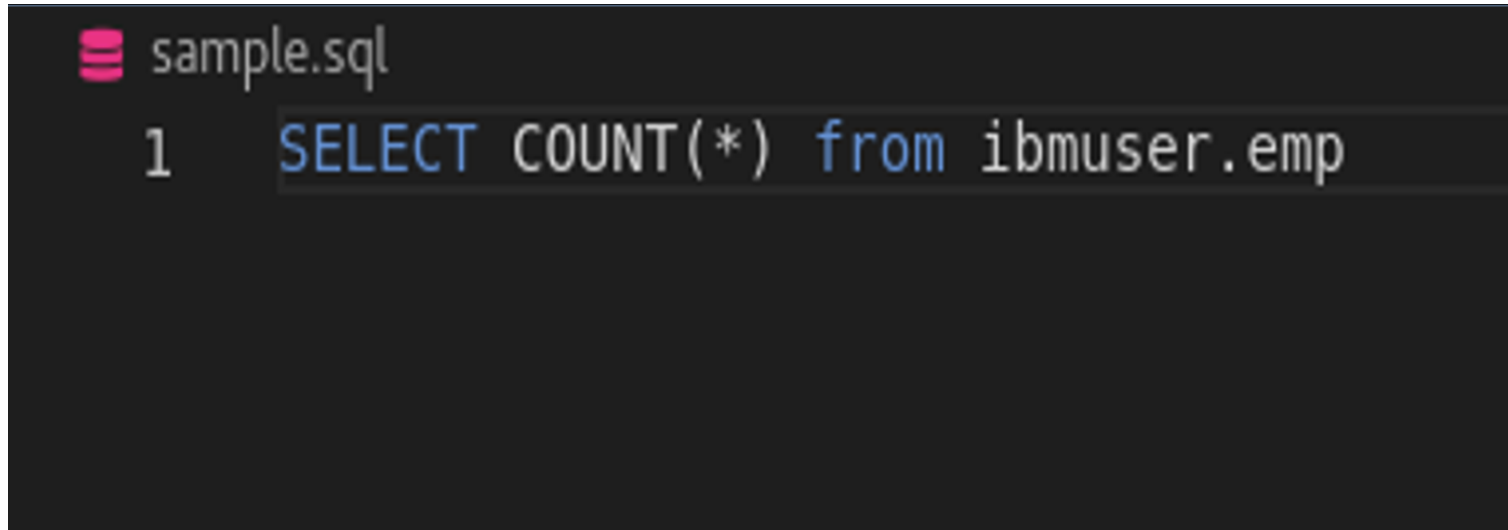
The **WHERE** clause is a filter that extracts only records that fulfill the specified condition.

Not only can you use strings, but you can use numbers to specify and other operators like “>” to filter by a greater-than range or “<=” to filter by a less-than-or-equal-to range.

SQL1250211-2139

9 MAKE IT COUNT (FOR REAL)

There are also some functions that allow you to count, average, or sum the rows - these are called “aggregate” functions.

A screenshot of a SQL editor with a dark background. At the top left, there is a file icon and the text 'sample.sql'. Below it, the number '1' is followed by the SQL query 'SELECT COUNT(*) from ibmuser.emp'. The query is highlighted with a light blue background.

```
sample.sql
1  SELECT COUNT(*) from ibmuser.emp
```

The syntax is pretty simple; you just write

- COUNT(column_name),
- AVG(column_name), or
- SUM(column_name)

after SELECT. There’s an example in the screenshot above.

Each of these functions takes the result of the query and performs the function on each row of the resulting column.

So if you add a WHERE to the query, it will only aggregate the amount of rows that meet the condition.

Now you are getting closer to turning normal questions someone might ask “*How many managers are there?*” into actual SQL queries.

“Can I make the display name of aggregate functions nicer?”

Yup! All you have to do is add whatever you want displayed as the name after the aggregate function.

Like this:

```
● SELECT COUNT(*) as NUM_EMPLOYEES from IBMUSER.EMP
```

Also something to note - these aggregate functions work on numerical and non-numerical columns, and they count all the rows in that column that have an entry.

When the asterisk is used in any of the functions, it counts the total number of rows.

10 ALL ABOUT JOINS

A “join” is the process of combining data from two or more tables based on some common domain of information.

Rows from one table are paired with rows from another table when information in the corresponding rows match based on the joining criteria.

In order to do this, a WHERE clause comes in handy again.

EMPNO	FIRSTNAME	MIDINIT	LASTNAME	WORKDEPT	PHONENO	HI
000010	CHRISTINE	I	HAAS	A00	3978	19/01
000020	MICHAEL	L	THOMPSON	B01	3476	19/10
000030	SALLY	A	KWAN	C01	4738	19/05
000050	JOHN	B	GEYER	F01	6789	19/27
000060	IRVING	F	STERN	D11	6423	19/14
000070	EVA	D	PULASKI	D21	7831	19/30
000090	EILEEN	W	HENDERSON	E11	5498	19/15
000100	THEODORE	O	SPENCER	E21	0972	19/29
000110	VINCENZO	G	LUCCHESE	A00	3490	19/19
000120	SEAN	O'CONNELL	A00	A00	2267	19/05
000130	DOLORES	M	QUINTANA	C01	4578	19/28
000140	HEATHER	A	NICHOLLS	C01	1793	19/15
000150	BRUCE		ADAMSON	D11	4510	19/12
000160	ELIZABETH	R	PLANKA	D11	3782	19/11
000170	MASATOSHI	J	YOSHIMURA	D11	3890	19/15

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
A00	SPOFFY COMPUTER SERVICE DEV.	000010	A00	
B01	PLANNING	000020	A00	
C01	INFORMATION CENTER	000030	A00	
D01	DEVELOPMENT CENTER	*NULL*	A00	
D11	MANUFACTURING SYSTEMS	000060	D01	
D21	ADMINISTRATION SYSTEMS	000070	D01	
E01	SUPPORT SERVICES	000050	A00	
E11	OPERATIONS	000090	F01	
E21	SOFTWARE SUPPORT	000100	E01	
F22	BRANCH OFFICE F2	*NULL*	E01	
G22	BRANCH OFFICE G2	*NULL*	F01	
H22	BRANCH OFFICE H2	*NULL*	F01	
I22	BRANCH OFFICE I2	*NULL*	E01	
J22	BRANCH OFFICE J2	*NULL*	E01	

You simply have to relate two common columns from two different tables by setting them equal to each other in the WHERE clause. The example above is from the DEPT and EMP tables; these tables can be joined on their common columns - WORKDEPT and DEPTNO.

Note that it is the **data values** in the columns that are common - not necessarily the column names.

SQL1250211-2139

11 JOINS HAPPEN

```
sample.sql
1  SELECT DEPTNAME, COUNT(EMPNO)
2     FROM ibmuser.dept D, ibmuser.emp E
3     WHERE D. DEPTNO = E.WORKDEPT
4     GROUP BY DEPTNAME
5     ORDER BY DEPTNAME
```

Run the example above.

Notice how this query joins IBMUSER.DEPT and IBMUSER.EMP on the columns DEPTNO and WORKDEPT. Also notice how there is a variable provided in the FROM clause that represents each table (**D** represents ibmuser.dept and **E** is for ibmuser.emp) – these are “correlation identifiers”.

There are a couple other syntax things to look at in detail after you run the query.

SQL1250211-2139

12 PURE FILTERED RESULTS

Status	Result 1
DEPTNAME	2
ADMINISTRATION SYSTEMS	7
INFORMATION CENTER	4
MANUFACTURING SYSTEMS	11
OPERATIONS	7

When you look at the result, it successfully counts the number of employees from each department.

In order to do this, the query had to use the **GROUP BY** keyword.

This clause groups rows that have the same values into summary rows. This is commonly used for aggregate functions (COUNT, SUM, AVG, MAX, MIN), as in the earlier example.

GROUP BY can be a little confusing, so experiment with the query to see what works, what does not, and what different combinations produce.

The last piece of the puzzle is the **ORDER BY** keyword; this simply displays the information in alphabetical order by the column specified.

SQL1250211-2139

“Can I join more than two tables?”

You can! This is called a “multi-table join”.

It is very similar in structure to the two-table join but you add an AND clause after WHERE that specifies a join to another table.

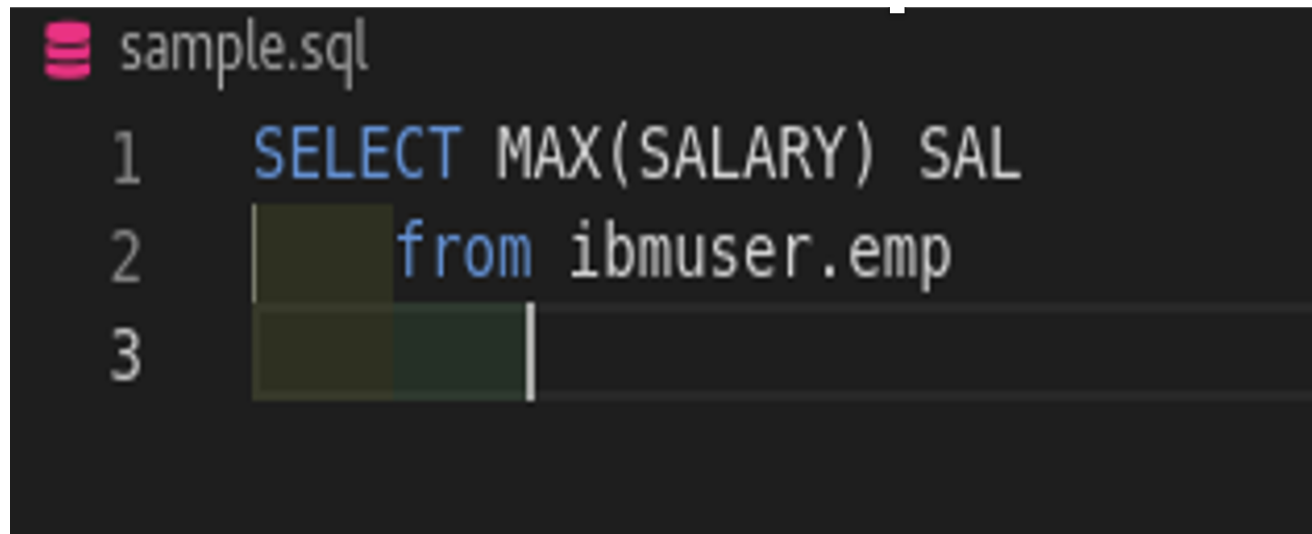
It looks like this:

```
SELECT company, stock_num, menu_code
FROM customer c, orders o, items i
WHERE c.customer_num = o.customer_num
      AND o.order_num = i.order_num;
```

13 PUSH IT TO THE MAX

One last, quick addition to your growing knowledge of SQL will be the **MIN** and **MAX** functions.

Just as they sound, these functions return the smallest value and the greatest value respectively from the specified columns. They are used in the same way COUNT, SUM, and AVG are used.

A screenshot of a SQL editor window titled 'sample.sql'. The window has a dark background with light-colored text. The SQL query is as follows:

```
1  SELECT MAX(SALARY) SAL
2  |      from ibmuser.emp
3  |      |
```

The cursor is positioned at the end of the third line. The text is color-coded: 'SELECT' is blue, 'MAX(SALARY)' is white, 'SAL' is blue, 'from' is blue, and 'ibmuser.emp' is white.

You can also name the output of these functions anything you want by adding a name immediately after the function clause as shown in the screenshot above.

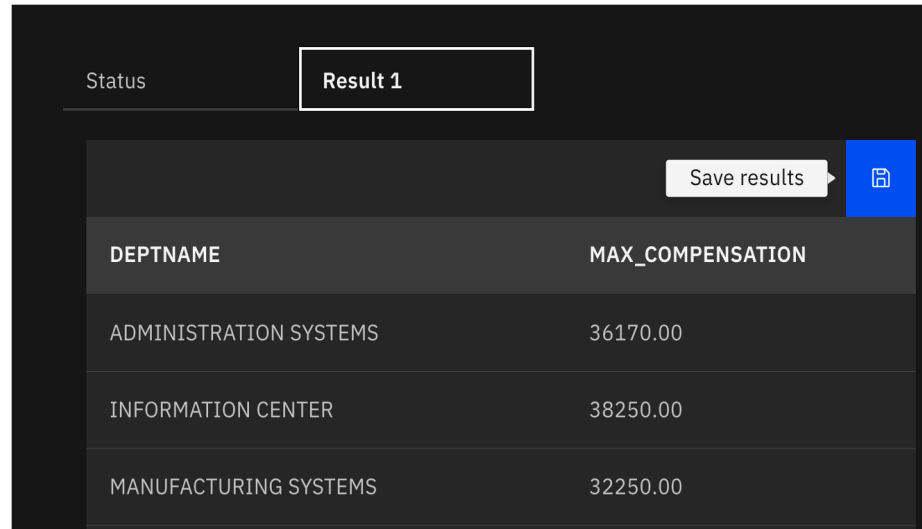
Again, play around with the functions to get some more familiarity before you move onto the challenge!

Experimenting on your own to observe the results of different combinations of syntax is one of the best ways you can learn!

14 PUTTING IT ALL TOGETHER

Your challenge is to query the maximum compensation (**all the ways** of paying someone) in each department sort by department name.

Display the results as two columns - **DEPTNAME** and **MAX_COMPENSATION**.



DEPTNAME	MAX_COMPENSATION
ADMINISTRATION SYSTEMS	36170.00
INFORMATION CENTER	38250.00
MANUFACTURING SYSTEMS	32250.00

Make sure you're thinking about all of the SQL syntax you have learned - you will need it!

Make sure you know what all of the columns in each table are; perhaps write a list of the columns in each table. When you have the output you want, download it as a **.csv** file by clicking the File icon in the results tab. This will download it to your laptop/workstation.

Hint: there is more than one column that counts as “compensation”!

SQL1250211-2139

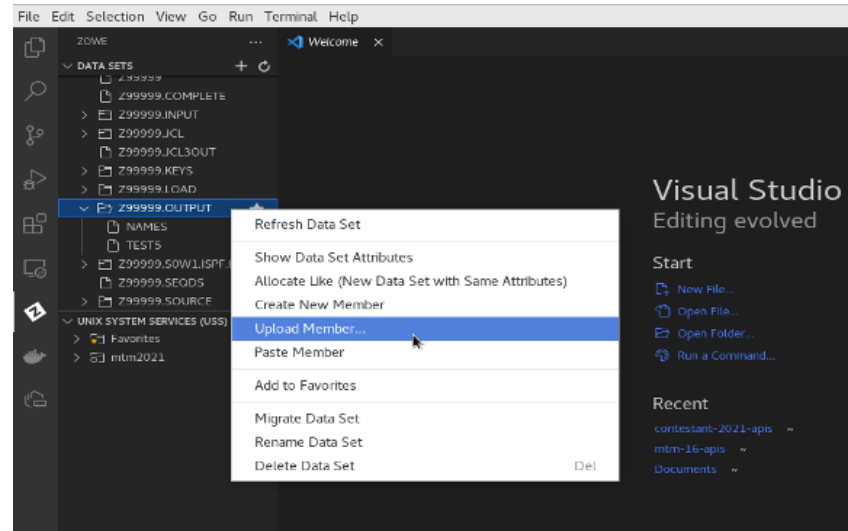
EMPNO	FIRSTNAME	MIDINIT	LASTNAME	WORKDEPT	PHONENO	HIREDATE	JOB	EDLEVEL	SEX	BIRTHDATE	SALARY	BONUS	COMM
000010	CHRISTINE	I	HAAS	A00	3978	1965-01-01	PRES	18	F	1933-08-14	52750.00	1000.00	4220.00
000015	CARLA	J	SOUSA	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*
000020	MICHAEL	L	THOMPSON	B01	3476	1973-10-10	MANAGER	18	M	1948-02-02	41250.00	800.00	3300.00
000025	LIVIA	L	TESTE	D21	2095	2021-09-09	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*
000030	SALLY	A	KWAN	C01	4738	1975-04-05	MANAGER	20	F	1941-05-11	38250.00	800.00	3060.00
000035	REBECA	R	TESTE	D21	2095	2021-09-09	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*
000050	JOHN	B	GEYER	E01	6789	1949-08-17	MANAGER	16	M	1925-09-15	40175.00	800.00	3214.00
000060	IRVING	F	STERN	D11	6423	1973-09-14	MANAGER	16	M	1945-07-07	32250.00	600.00	2580.00

- **COMM** is the “commission” column - it shows the amount an employee has been paid from the value of sales.
- **BONUS** shows how much an employee has been paid because they have achieved targets, or for winning sales competitions.

SQL1250211-2139

15 LAST CHECK AND WRAP-UP

It's now time to validate the challenge, so after you've downloaded the **.csv** file, right-click on your **Zxxxxx.OUTPUT** data set and click on "Upload Member".



Then upload your **.csv** file. After uploading, rename the file as **DB2OUT** so the checker job can find it, and you should be good to go!

Now it's time to run the check to see if your query was correct. Submit the **CHKSQL** member from **ZXP.PUBLIC.JCL** and look for a completion code of 0000!

Nice job - let's recap	Next up ...
<p>You just learned a ton of new information about relational databases and SQL syntax. Keep in mind, Db2 is just one relational database system - there are many more, but SQL stays the same!</p> <p>So you now know how to query all kinds of information from tables, as well as how to join tables together so you can relate information across different tables.</p> <p>Pretty powerful stuff!</p>	<p>Check out the other available Advanced challenges</p>