#4IR

Techma Zone

SQL Assignment

By: Yuaan Hussain

www.linkedin.com/in/muneer-ahmed-quadri

Database # 1

```
database1 = 'ConstructCo.db'
conn = sql.connect(database1)
```

```
def read_query(q):
    return pd.read_sql_query(q, conn)
```

```
read_query('''select * from sqlite_master where type == "table"''')
```

	type	name	tbl_name	rootpage	sql
0	table	JOB	JOB	2	CREATE TABLE JOB (\n JOB_CODE VARCHAR(...
1	table	EMPLOYEE	EMPLOYEE	4	CREATE TABLE EMPLOYEE (\n EMP_NUM VARCH...
2	table	PROJECT	PROJECT	6	CREATE TABLE PROJECT (\n PROJ_NUM VARCHA...
3	table	ASSIGNMENT	ASSIGNMENT	8	CREATE TABLE ASSIGNMENT (\n ASSIGN_NUM I...
4	table	EMP1	EMP1	10	CREATE TABLE EMP1 (\n EMP_NUM VARCHAR(3) PR...
5	table	EMP2	EMP2	12	CREATE TABLE EMP2(\n EMP_NUM TEXT,\n EMP_LNA...

Database # 2

```
database2 = 'SaleCo.db'
conn2 = sql.connect(database2)
```

```
def read_query2(q):
    return pd.read_sql_query(q, conn2)
```

```
read_query2('''select * from sqlite_master where type == "table"''')
```

	type	name	tbl_name	rootpage	sql
0	table	VENDOR	VENDOR	2	CREATE TABLE VENDOR (\nV_CODE \t\tINTEGER,\nV_...
1	table	PRODUCT	PRODUCT	3	CREATE TABLE PRODUCT (\nP_CODE \tVARCHAR(10) P...
2	table	CUSTOMER	CUSTOMER	5	CREATE TABLE CUSTOMER (\nCUS_CODE\tINTEGER PRI...
3	table	INVOICE	INVOICE	7	CREATE TABLE INVOICE (\nINV_NUMBER \tINTEG...
4	table	LINE	LINE	8	CREATE TABLE LINE (\nINV_NUMBER \tINTEGER NOT ...
5	table	EMPLOYEE	EMPLOYEE	11	CREATE TABLE EMPLOYEE (\nEMP_NUM\t\tINTEGER\tP...
6	table	EMP	EMP	12	CREATE TABLE EMP (\nEMP_NUM\t\tINTEGER\tPRIMAR...

P1. Write the SQL code required to list the employee number, last name, first name, and middle initial of all employees whose last names start with Smith. In other words, the rows for both Smith and Smithfield should be included in the listing. Sort the results by employee number. Assume case sensitivity.

```
read_query("""SELECT EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL
from EMPLOYEE
where EMP_LNAME LIKE 'Smith%'
order by EMP_NUM
""")
```

	EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL
0	106	Smithfield	William	None
1	109	Smith	Larry	W
2	112	Smithson	Darlene	M

P2. Using the EMPLOYEE, JOB, and PROJECT tables in the ConstructCo database, write the SQL code that will join the EMPLOYEE and PROJECT tables using EMP NUM as the common attribute. Display the attributes shown in the results presented in Figure 2, sorted by project value.

```
read_query("""select p.PROJ_NAME, p.PROJ_VALUE, p.PROJ_BALANCE, e.EMP_LNAME,
e.EMP_FNAME, e.EMP_INITIAL, j.JOB_CODE, j.JOB_DESCRIPTION, j.JOB_CHG_HOUR
from JOB j inner join EMPLOYEE e
on j.JOB_CODE = e.JOB_CODE
inner join PROJECT p
on e.EMP_NUM = p.EMP_NUM
order by p.PROJ_VALUE
""")
```

	PROJ_NAME	PROJ_VALUE	PROJ_BALANCE	EMP_LNAME	EMP_FNAME	EMP_INITIAL	JOB_CODE	JOB_DESCRIPTION	JOB_CHG_HOUR
0	Rolling Tide	805000	500345.2	Senior	David	H	501	Systems Analyst	96.75
1	Evergreen	1453500	1002350.0	Arbough	June	E	500	Programmer	35.75
2	Starflight	2650500	2309880.0	Alonzo	Maria	D	500	Programmer	35.75
3	Amber Wave	3500500	2110346.0	Washington	Ralph	B	501	Systems Analyst	96.75

P3. Write the SQL code that will produce the same information that was shown in Problem P2., but sorted by the employee's last name.

```
read_query("""select p.PROJ_VALUE, p.PROJ_BALANCE, e.EMP_LNAME, e.EMP_FNAME,
e.EMP_INITIAL, j.JOB_CODE, j.JOB_DESCRIPTION, j.JOB_CHG_HOUR
from JOB j inner join EMPLOYEE e
on j.JOB_CODE = e.JOB_CODE
inner join PROJECT p
on e.EMP_NUM = p.EMP_NUM
order by e.EMP_LNAME
""")
```

	PROJ_VALUE	PROJ_BALANCE	EMP_LNAME	EMP_FNAME	EMP_INITIAL	JOB_CODE	JOB_DESCRIPTION	JOB_CHG_HOUR
0	2650500	2309880.0	Alonzo	Maria	D	500	Programmer	35.75
1	1453500	1002350.0	Arbough	June	E	500	Programmer	35.75
2	805000	500345.2	Senior	David	H	501	Systems Analyst	96.75
3	3500500	2110346.0	Washington	Ralph	B	501	Systems Analyst	96.75

P4. Write the SQL code that will list only the distinct project numbers in the ASSIGNMENT table, sorted by project number.

```
read_query("""select DISTINCT PROJ_NAME
from PROJECT
order by PROJ_NUM
""")
```

	PROJ_NAME
0	Evergreen
1	Amber Wave
2	Rolling Tide
3	Starflight

P5. Write the SQL code to validate the ASSIGN CHARGE values in the ASSIGNMENT table. Your query should retrieve the assignment number, employee number, project number, the stored assignment charge (ASSIGN CHARGE), and the calculated assignment charge (calculated by multiplying ASSIGN CHG HR by ASSIGN HOURS). Sort the results by the assignment number.

```
read_query("""select a.ASSIGN_NUM, e.EMP_NUM, p.PROJ_NUM, a.ASSIGN_CHARGE,
a.ASSIGN_CHG_HR * ASSIGN_HOURS 'CALCULATED ASSIGNMENT CHARGE'
from EMPlOYEE e inner join PROJECT p
on e.EMP_NUM = p.EMP_NUM
inner join ASSIGNMENT a
on p.EMP_NUM = a.EMP_NUM
order by ASSIGN_NUM
""")
```

	ASSIGN_NUM	EMP_NUM	PROJ_NUM	ASSIGN_CHARGE	CALCULATED ASSIGNMENT CHARGE
0	1001	103	15	295.75	295.75
1	1004	103	15	498.55	498.55
2	1005	108	18	212.85	212.85
3	1008	103	15	76.05	76.05
4	1012	108	18	328.95	328.95
5	1015	103	15	515.45	515.45
6	1021	108	18	298.35	298.35
7	1024	103	15	278.85	278.85

P6. Using the data in the ASSIGNMENT table, write the SQL code that will yield the total number of hours worked for each employee and the total charges stemming from those hours worked, sorted by employee number. The results of running that query are shown in Figure 3.

```
read_query("""select DISTINCT a.EMP_NUM, e.EMP_LNAME, sum(a.ASSIGN_HOURS) 'Sum of ASSIGN_HOURS',
sum(a.ASSIGN_CHARGE) 'Sum of ASSIGN_CHARGE'
from EMPLOYEE e inner join ASSIGNMENT a
on e.EMP_NUM = a.EMP_NUM
group by a.EMP_NUM
order by a.EMP_NUM
""")
```

	EMP_NUM	EMP_LNAME	Sum of ASSIGN_HOURS	Sum of ASSIGN_CHARGE
0	101	News	3.1	387.50
1	103	Arbough	19.7	1664.65
2	104	Ramoras	11.9	1218.70
3	105	Johnson	12.5	1382.50
4	108	Washington	8.3	840.15
5	113	Joenbrood	3.8	192.85
6	115	Bawangi	12.5	1276.75
7	117	Williamson	18.8	649.54

P7. Write a query to produce the total number of hours and charges for each of the projects represented in the ASSIGNMENT table, sorted by project number. The output is shown in Figure 4.

```
read_query("""select PROJ_NUM, sum(ASSIGN_HOURS) 'Sum of ASSIGN_HOURS',
sum(ASSIGN_CHARGE) 'Sum of ASSIGN_CHARGE'
from ASSIGNMENT
group by PROJ_NUM
order by PROJ_NUM
""")
```

	PROJ_NUM	Sum of ASSIGN_HOURS	Sum of ASSIGN_CHARGE
0	15	20.5	1806.52
1	18	23.7	1544.80
2	22	27.0	2593.16
3	25	19.4	1668.16

P8. Write the SQL code to generate the total hours worked and the total charges made by all employees. The results are shown in Figure 5.

```
read_query("""select sum(ASSIGN_HOURS) 'Sum of Sum of ASSIGN_HOURS',
sum(ASSIGN_CHARGE) 'Sum of Sum of ASSIGN_CHARGE'
from ASSIGNMENT
""")
```

	Sum of Sum of ASSIGN_HOURS	Sum of Sum of ASSIGN_CHARGE
0	90.6	7612.64

P9. Write a query to count the number of invoices.

```
read_query2('select count(INV_NUMBER) from INVOICE')
```

	count(INV_NUMBER)
0	8

P10. Write a query to count the number of customers with a balance of more than $500.

```
read_query2("select count(CUS_CODE) 'Number of Customers' from CUSTOMER where CUS_BALANCE>500")
```

	Number of Customers
0	2

P11. Using the output shown in Figure 7 as your guide, generate a list of customer purchases, including the subtotals for each of the invoice line numbers. The subtotal is a derived attribute calculated by multiplying LINE UNITS by LINE PRICE. Sort the output by customer code, invoice number, and product description. Be certain to use the column aliases as shown in the figure.

```
read_query2("""select c.CUS_CODE, i.INV_NUMBER, p.P_DESCRIPT, l.LINE_UNITS 'Units Bought',
l.LINE_PRICE 'Units Price', l.LINE_UNITS * LINE_PRICE 'Subtotal'
from CUSTOMER c inner join INVOICE i
on c.CUS_CODE = i.CUS_CODE
inner join LINE l
on i.INV_NUMBER = l.INV_NUMBER
inner join PRODUCT p
on l.P_CODE = p.P_CODE
order by c.CUS_CODE, i.INV_NUMBER, p.P_DESCRIPT
""")
```

	CUS_CODE	INV_NUMBER	P_DESCRIPT	Units Bought	Units Price	Subtotal
0	10011	1002	Rat-tail file, 1/8-in. fine	2	4.99	9.98
1	10011	1004	Claw hammer	2	9.95	19.90
2	10011	1004	Rat-tail file, 1/8-in. fine	3	4.99	14.97
3	10011	1008	Claw hammer	1	9.95	9.95
4	10011	1008	PVC pipe, 3.5-in., 8-ft	5	5.87	29.35
5	10011	1008	Steel matting, 4'x8'x1/6", .5" mesh	3	119.95	359.85
6	10012	1003	7.25-in. pwr. saw blade	5	14.99	74.95
7	10012	1003	B&D cordless drill, 1/2-in.	1	38.95	38.95
8	10012	1003	Hrd. cloth, 1/4-in., 2x50	1	39.95	39.95
9	10014	1001	7.25-in. pwr. saw blade	1	14.99	14.99
10	10014	1001	Claw hammer	1	9.95	9.95
11	10014	1006	1.25-in. metal screw, 25	3	6.99	20.97
12	10014	1006	B&D jigsaw, 12-in. blade	1	109.92	109.92
13	10014	1006	Claw hammer	1	9.95	9.95
14	10014	1006	Hicut chain saw, 16 in.	1	256.99	256.99
15	10015	1007	7.25-in. pwr. saw blade	2	14.99	29.98
16	10015	1007	Rat-tail file, 1/8-in. fine	1	4.99	4.99
17	10018	1005	PVC pipe, 3.5-in., 8-ft	12	5.87	70.44

P12. Write a query to display the customer code, balance, and total purchases for each customer. Total purchase is calculated by summing the line subtotals (as calculated in Problem P11.) for each customer. Sort the results by customer code, and use aliases as shown in Figure 8.

```
read_query2("""select c.CUS_CODE, c.CUS_BALANCE, sum(l.LINE_PRICE * LINE_UNITS) 'Total Purchase'
from CUSTOMER c inner join INVOICE i
on c.CUS_CODE = i.CUS_CODE
inner join LINE l
on i.INV_NUMBER = l.INV_NUMBER
group by c.CUS_CODE
order by c.CUS_CODE
""")
```

	CUS_CODE	CUS_BALANCE	Total Purchase
0	10011	0.00	444.00
1	10012	345.86	153.85
2	10014	0.00	422.77
3	10015	0.00	34.97
4	10018	216.55	70.44

P13. Modify the query in Problem P12. to include the number of individual product purchases made by each customer. (In other words, if the customer's invoice is based on three products, one per LINE NUMBER, you count three product purchases. Note that in the original invoice data, customer 10011 generated three invoices, which contained a total of six lines, each representing a product purchase.) Your output values must match those shown in Figure 9, sorted by customer code.

```
read_query2("""select c.CUS_CODE, c.CUS_BALANCE, sum(l.LINE_PRICE * LINE_UNITS) 'Total Purchase',
count(LINE_NUMBER) 'Number of Purchase'
from CUSTOMER c inner join INVOICE i
on c.CUS_CODE = i.CUS_CODE
inner join LINE l
on i.INV_NUMBER = l.INV_NUMBER
group by c.CUS_CODE
order by c.CUS_CODE
""")
```

	CUS_CODE	CUS_BALANCE	Total Purchase	Number of Purchase
0	10011	0.00	444.00	6
1	10012	345.86	153.85	3
2	10014	0.00	422.77	6
3	10015	0.00	34.97	2
4	10018	216.55	70.44	1

P14. Use a query to compute the total of all purchases, the number of purchases, and the average purchase amount made by each customer. Your output values must match those shown in Figure 10. Sort the results by customer code.

```
read_query2("""select c.CUS_CODE, c.CUS_BALANCE, sum(l.LINE_PRICE * LINE_UNITS) 'Total Purchase',
count(LINE_NUMBER) 'Number of Purchase', round(avg(l.LINE_PRICE * LINE_UNITS), 2) 'Average Purchase Amount'
from CUSTOMER c inner join INVOICE i
on c.CUS_CODE = i.CUS_CODE
inner join LINE l
on i.INV_NUMBER = l.INV_NUMBER
group by c.CUS_CODE
order by c.CUS_CODE
""")
```

	CUS_CODE	CUS_BALANCE	Total Purchase	Number of Purchase	Average Purchase Amount
0	10011	0.00	444.00	6	74.00
1	10012	345.86	153.85	3	51.28
2	10014	0.00	422.77	6	70.46
3	10015	0.00	34.97	2	17.49
4	10018	216.55	70.44	1	70.44

P15. Create a query to produce the total purchase per invoice, generating the results shown in Figure 11, sorted by invoice number. The invoice total is the sum of the product purchases in the LINE that corresponds to the INVOICE.

```
read_query2("""select INV_NUMBER, sum(LINE_PRICE * lINE_UNITS) 'Invoice Total'
from LINE
group by INV_NUMBER
order by INV_NUMBER
""")
```

	INV_NUMBER	Invoice Total
0	1001	24.94
1	1002	9.98
2	1003	153.85
3	1004	34.87
4	1005	70.44
5	1006	397.83
6	1007	34.97
7	1008	399.15

P16. Use a query to show the invoices and invoice totals in Figure 12. Sort the results by customer code and then by invoice number.

```
read_query2("""select i.CUS_CODE, l.INV_NUMBER, sum(l.LINE_PRICE * LINE_UNITS) 'INVOICE TOTAL'
from INVOICE i inner join LINE l
on i.INV_NUMBER = l.INV_NUMBER
group by l.INV_NUMBER
order by 1, 2
""")
```

	CUS_CODE	INV_NUMBER	INVOICE TOTAL
0	10011	1002	9.98
1	10011	1004	34.87
2	10011	1008	399.15
3	10012	1003	153.85
4	10014	1001	24.94
5	10014	1006	397.83
6	10015	1007	34.97
7	10018	1005	70.44

P17. Write a query to produce the number of invoices and the total purchase amounts by customer, using the output shown in Figure 13 as your guide. Note the results are sorted by customer code. (Compare this summary to the results shown in Problem P16..)

```
read_query2("""select i.CUS_CODE, count(l.INV_NUMBER) 'No of Invoices',
sum(l.LINE_UNITS * LINE_PRICE) 'Total Customer Purchase'
from INVOICE i inner join LINE l
on i.INV_NUMBER = l.INV_NUMBER
group by i.CUS_CODE
order by i.CUS_CODE
""")
```

	CUS_CODE	No of Invoices	Total Customer Purchase
0	10011	6	444.00
1	10012	3	153.85
2	10014	6	422.77
3	10015	2	34.97
4	10018	1	70.44

P18. Write a query to generate the total number of invoices, the invoice total for all of the invoices, the smallest of the customer purchase amounts, the largest of the customer purchase amounts, and the average of all the customer purchase amounts. Your output must match Figure 14.

```
read_query2('''select
count(INV_NUMBER) 'Total Invoice',
sum(TOTAL_SALES) 'Total Sales',
min(TOTAL_SALES) 'Min Customer Purchase',
max(TOTAL_SALES) 'Largest Customer Purchase',
avg(TOTAL_SALES) 'Avg Customer Purchase'
from (select l.INV_NUMBER, sum(l.LINE_UNITS * l.LINE_PRICE) 'TOTAL_SALES'
from CUSTOMER c inner join INVOICE i
on i.CUS_CODE = c.CUS_CODE
inner join LINE l
on i.INV_NUMBER = l.INV_NUMBER
group by c.CUS_CODE) 'InvoiceSales' ''')
```

	Total Invoice	Total Sales	Min Customer Purchase	Largest Customer Purchase	Avg Customer Purchase
0	5	1126.03	34.97	444.0	225.206

P19. List the balances of customers who have made purchases during the current invoice cycle – that is, for the customers who appear in the INVOICE table. The results of this query are shown in Figure 15, sorted by customer code.

```
read_query2("""select DISTINCT i.CUS_CODE, c.CUS_BALANCE
from CUSTOMER c inner join INVOICE i
on c.CUS_CODE = i.CUS_CODE
order by 1
""")
```

	CUS_CODE	CUS_BALANCE
0	10011	0.00
1	10012	345.86
2	10014	0.00
3	10015	0.00
4	10018	216.55

P20. Provide a summary of customer balance characteristics for customers who made purchases. Include the minimum balance, maximum balance, and average balance, as shown in Figure 16.

```
read_query2('''select min(c.CUS_BALANCE) 'Minimum Balance',
max(c.CUS_BALANCE) 'Maximum Balance',
(sum(c.CUS_BALANCE)/COUNT(DISTINCT c.CUS_CODE)) 'Avg Balance'
from CUSTOMER c inner join INVOICE i
ON i.CUS_CODE = c.CUS_CODE
order by c.CUS_CODE''')
```

	Minimum Balance	Maximum Balance	Avg Balance
0	0	345.86	112.482

P21. Create a query to find the balance characteristics for all customers, including the total of the outstanding balances. The results of this query are shown in Figure 17.

```
read_query2("""select sum(CUS_BALANCE) 'Total Balance', min(CUS_BALANCE) 'Minimum Balance',
max(CUS_BALANCE) 'Maximum Balance', avg(CUS_BALANCE) 'Avg Balance'
from CUSTOMER
""")
```

	Total Balance	Minimum Balance	Maximum Balance	Avg Balance
0	2089.28	0	768.93	208.928

P22. Find the listing of customers who did not make purchases during the invoicing period. Sort the results by customer code. Your output must match the output shown in Figure 18.

```
read_query2('''select DISTINCT c.CUS_CODE, c.CUS_BALANCE
from CUSTOMER c LEFT JOIN INVOICE i
on c.CUS_CODE = i.CUS_CODE
where i.CUS_CODE is null
order by c.CUS_CODE''')
```

	CUS_CODE	CUS_BALANCE
0	10010	0.00
1	10013	536.75
2	10016	221.19
3	10017	768.93
4	10019	0.00

P23. Find the customer balance summary for all customers who have not made purchases during the current invoicing period. The results are shown in Figure 19.

```
read_query2('''select sum(c.CUS_BALANCE) 'Total Balance',
min(c.CUS_BALANCE) 'Minimum Balance' ,
max(c.CUS_BALANCE) 'Maximum Balance',
avg(c.CUS_BALANCE) 'Avg Balance'
from CUSTOMER c left join INVOICE i
on c.CUS_CODE = i.CUS_CODE
where i.CUS_CODE is null
order by c.CUS_CODE''')
```

	Total Balance	Minimum Balance	Maximum Balance	Avg Balance
0	1526.87	0	768.93	305.374

P24. Create a query that summarizes the value of products currently in inventory. Note that the value of each product is a result of multiplying the units currently in inventory by the unit price. Sort the results in descending order by subtotal, as shown in Figure 20.

```
read_query2("""select P_DESCRIPT, P_QOH, P_PRICE, P_QOH * P_PRICE 'Sub Total'
from PRODUCT
order by 4 DESC
""")
```

	P_DESCRIPT	P_QOH	P_PRICE	Sub Total
0	Hicut chain saw, 16 in.	11	256.99	2826.89
1	Steel matting, 4'x8'x1/6", .5" mesh	18	119.95	2159.10
2	2.5-in. wd. screw, 50	237	8.45	2002.65
3	1.25-in. metal screw, 25	172	6.99	1202.28
4	PVC pipe, 3.5-in., 8-ft	188	5.87	1103.56
5	Hrd. cloth, 1/2-in., 3x50	23	43.99	1011.77
6	Power painter, 15 psi., 3-nozzle	8	109.99	879.92
7	B&D jigsaw, 12-in. blade	8	109.92	879.36
8	Hrd. cloth, 1/4-in., 2x50	15	39.95	599.25
9	B&D jigsaw, 8-in. blade	6	99.87	599.22
10	7.25-in. pwr. saw blade	32	14.99	479.68
11	B&D cordless drill, 1/2-in.	12	38.95	467.40
12	9.00-in. pwr. saw blade	18	17.49	314.82
13	Claw hammer	23	9.95	228.85
14	Rat-tail file, 1/8-in. fine	43	4.99	214.57
15	Sledge hammer, 12 lb.	8	14.40	115.20

P25. Find the total value of the product inventory. The results are shown in Figure 21.

```
read_query2("""select sum(P_QOH * P_PRICE) 'Total value of Inventory'
from PRODUCT
""")
```

	Total value of Inventory
0	15084.52