

1. Create a table "Station" to store information about weather observation stations

```
create table Station
(ID INT PRIMARY KEY,
 CITY CHAR(20),
 STATE CHAR(2),
 LAT_N INT,
 LONG_W INT)
```

```
1 v create table Station
2 (ID INT PRIMARY KEY,
3   CITY CHAR(20),
4   STATE CHAR(2),
5   LAT_N INT,
6   LONG_W INT)
7
8
```

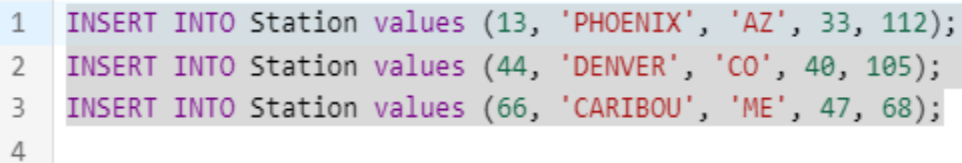
Table created.

2. Insert the following records into the table:

```
INSERT INTO Station values (13, 'PHOENIX', 'AZ', 33, 112);
```

```
INSERT INTO Station values (44, 'DENVER', 'CO', 40, 105);
```

```
INSERT INTO Station values (66, 'CARIBOU', 'ME', 47, 68);
```

A screenshot of a SQL editor window. The editor has a light gray background with a vertical line number column on the left. The first three lines of code are highlighted in a light blue color. The code consists of three SQL INSERT statements. The first line is '1 INSERT INTO Station values (13, 'PHOENIX', 'AZ', 33, 112);'. The second line is '2 INSERT INTO Station values (44, 'DENVER', 'CO', 40, 105);'. The third line is '3 INSERT INTO Station values (66, 'CARIBOU', 'ME', 47, 68);'. The fourth line is '4' and is empty. The editor window has a dark gray title bar at the top.

```
1 INSERT INTO Station values (13, 'PHOENIX', 'AZ', 33, 112);  
2 INSERT INTO Station values (44, 'DENVER', 'CO', 40, 105);  
3 INSERT INTO Station values (66, 'CARIBOU', 'ME', 47, 68);  
4
```

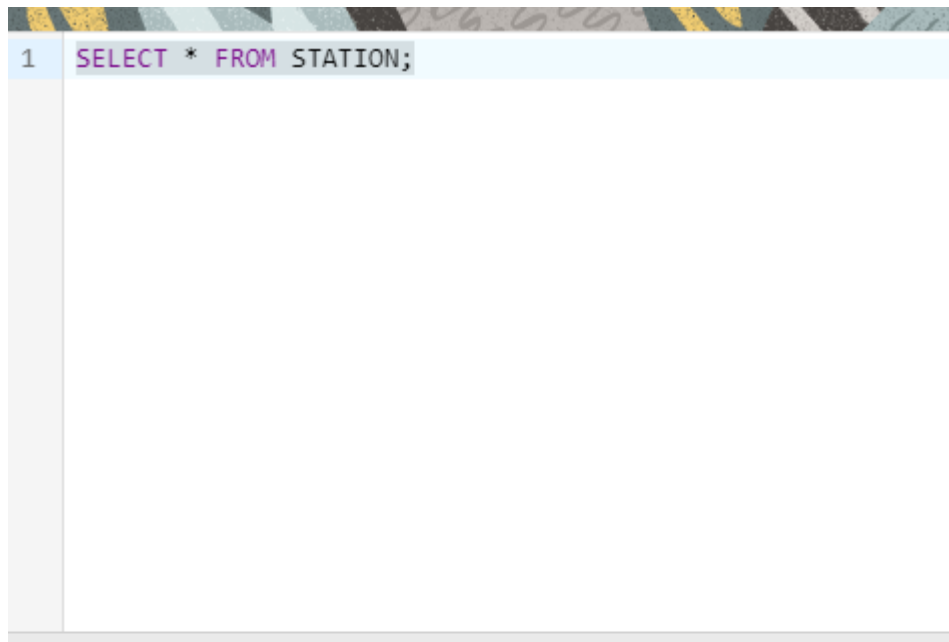
```
1 row(s) inserted.
```

```
1 row(s) inserted.
```

```
1 row(s) inserted.
```

3. Execute a query to look at table STATION in undefined order.

```
SELECT * FROM STATION;
```



ID	CITY	STATE	LAT_N	LONG_W
13	PHOENIX	AZ	33	112
44	DENVER	CO	40	105
66	CARIBOU	ME	47	68

Download CSV

3 rows selected.

4. Execute a query to select Northern stations (Northern latitude > 39.7).

```
SELECT * FROM STATION  
WHERE LAT_N > 39.7;
```

```
1 v SELECT * FROM STATION  
2 WHERE LAT_N > 39.7;  
3
```

ID	CITY	STATE	LAT_N	LONG_W
44	DENVER	CO	40	105
66	CARIBOU	ME	47	68

Download CSV

2 rows selected.

5. Create another table, 'STATS', to store normalized temperature and precipitation data:

```
CREATE TABLE STATS (  
  
  ID INT,  
  
  "MONTH" INT CHECK ("MONTH" BETWEEN 1 AND 12),  
  
  TEMP_F DECIMAL(5, 2) CHECK (TEMP_F BETWEEN -80 AND 150),  
  
  RAIN_I DECIMAL(5, 2) CHECK (RAIN_I BETWEEN 0 AND 100),  
  
  FOREIGN KEY (ID) REFERENCES STATION(ID),  
  
  CONSTRAINT UC_ID_MONTH UNIQUE (ID, "MONTH")  
  
);
```

```
1 v CREATE TABLE STATS (  
2   ID INT,  
3   "MONTH" INT CHECK ("MONTH" BETWEEN 1 AND 12),  
4   TEMP_F DECIMAL(5, 2) CHECK (TEMP_F BETWEEN -80 AND 150),  
5   RAIN_I DECIMAL(5, 2) CHECK (RAIN_I BETWEEN 0 AND 100),  
6   FOREIGN KEY (ID) REFERENCES STATION(ID),  
7   CONSTRAINT UC_ID_MONTH UNIQUE (ID, "MONTH")  
8 );  
9  
10  
11
```

Table created.

6. Populate the table STATS with some statistics for January and July:

```
INSERT INTO STATS VALUES
```

```
(13, 1, 57.4, .31);
```

```
INSERT INTO STATS VALUES
```

```
(13, 7, 91.7, 5.15);
```

```
INSERT INTO STATS VALUES
```

```
(44, 1, 27.3, .18);
```

```
INSERT INTO STATS VALUES
```

```
(44, 7, 74.8, 2.11);
```

```
INSERT INTO STATS VALUES
```

```
(66, 1, 6.7, 2.1);
```

```
INSERT INTO STATS VALUES
```

```
(66, 7, 65.8, 4.52);
```

```
1 v INSERT INTO STATS VALUES
2 (13, 1, 57.4, .31);
3 v INSERT INTO STATS VALUES
4 (13, 7, 91.7, 5.15);
5 v INSERT INTO STATS VALUES
6 (44, 1, 27.3, .18);
7 v INSERT INTO STATS VALUES
8 (44, 7, 74.8, 2.11);
9 v INSERT INTO STATS VALUES
10 (66, 1, 6.7, 2.1);
11 v INSERT INTO STATS VALUES
12 (66, 7, 65.8, 4.52);
13
14
15
```

```
1 row(s) inserted.
```

```
1 row(s) inserted.
```

```
1 row(s) inserted.
```

```
1 row(s) inserted.
```

```
1 row(s) inserted.
```

```
1 row(s) inserted.
```

7. Execute a query to display temperature stats (from STATS table) for each city (from Station table).

```
SELECT ST.CITY, AVG(S.TEMP_F) AS AVERAGE_TEMPERATURE, MIN(S.TEMP_F)
      AS MIN_TEMPERATURE, MAX(S.TEMP_F) AS MAX_TEMPERATURE
FROM STATS S JOIN STATION ST ON S.ID = ST.ID
GROUP BY ST.CITY;
```

```
1 | SELECT ST.CITY, AVG(S.TEMP_F) AS AVERAGE_TEMPERATURE, MIN(S.TEMP_F)
2 |     AS MIN_TEMPERATURE, MAX(S.TEMP_F) AS MAX_TEMPERATURE
3 | FROM STATS S JOIN STATION ST ON S.ID = ST.ID
4 | GROUP BY ST.CITY;
5 |
6 |
```

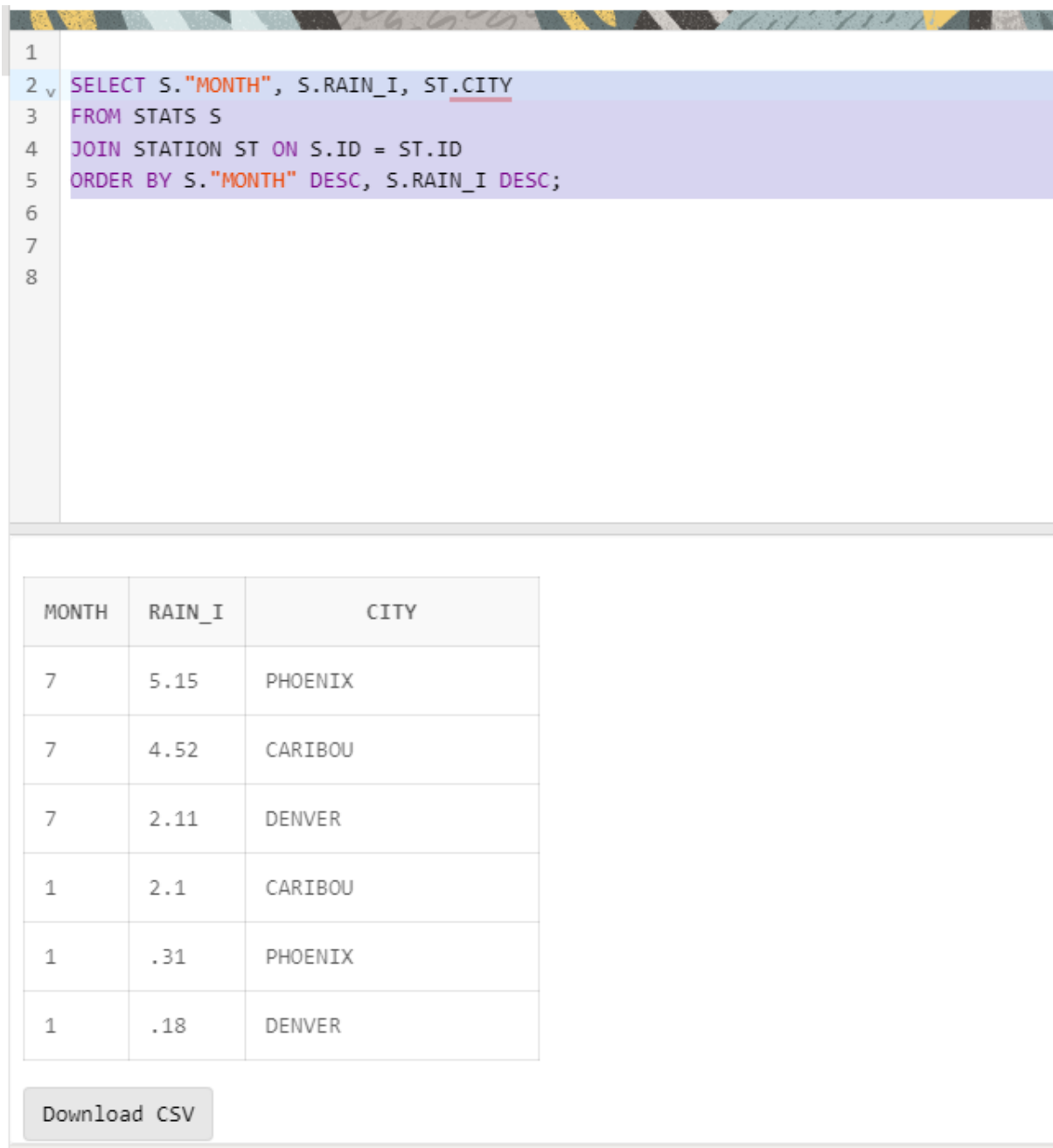
CITY	AVERAGE_TEMPERATURE	MIN_TEMPERATURE	MAX_TEMPERATURE
CARIBOU	36.25	6.7	65.8
DENVER	51.05	27.3	74.8
PHOENIX	74.55	57.4	91.7

Download CSV

3 rows selected.

8. Execute a query to look at the table STATS, ordered by month and greatest rainfall, with columns rearranged. It should also show the corresponding cities.

```
SELECT S."MONTH", S.RAIN_I, ST.CITY  
  
FROM STATS S  
  
JOIN STATION ST ON S.ID = ST.ID  
  
ORDER BY S."MONTH" DESC, S.RAIN_I DESC;
```



The screenshot shows a SQL query editor with a line number column on the left (1-8) and a query text area. The query is: `SELECT S."MONTH", S.RAIN_I, ST.CITY FROM STATS S JOIN STATION ST ON S.ID = ST.ID ORDER BY S."MONTH" DESC, S.RAIN_I DESC;`. Below the editor is a table with 3 columns: MONTH, RAIN_I, and CITY. The table contains 6 rows of data. At the bottom left, there is a button labeled "Download CSV".

MONTH	RAIN_I	CITY
7	5.15	PHOENIX
7	4.52	CARIBOU
7	2.11	DENVER
1	2.1	CARIBOU
1	.31	PHOENIX
1	.18	DENVER

Download CSV

9. Execute a query to look at temperatures for July from table STATS, lowest temperatures first, picking up city name and latitude.

```
SELECT S."MONTH", S.TEMP_F, ST.CITY, ST.LAT_N
FROM STATS S
JOIN STATION ST ON S.ID = ST.ID
WHERE "MONTH"=7
ORDER BY S.TEMP_F;
```

```
1 v SELECT S."MONTH", S.TEMP_F, ST.CITY, ST.LAT_N FROM STATS S
2     JOIN STATION ST
3     ON S.ID = ST.ID
4 WHERE "MONTH" = 7
5     ORDER BY S.TEMP_F;
6
7
```

MONTH	TEMP_F	CITY	LAT_N
7	65.8	CARIBOU	47
7	74.8	DENVER	40
7	91.7	PHOENIX	33

Download CSV

3 rows selected.

10. Execute a query to show MAX and MIN temperatures as well as average rainfall for each city.

```
SELECT ST.CITY, MAX(S.TEMP_F) AS MAX_Temperature,  
       MIN(S.TEMP_F) AS MIN_Temperature,  
       AVG(S.TEMP_F) AS AVERAGE_Temperature FROM STATS S  
JOIN STATION ST  
ON S.ID = ST.ID  
GROUP BY CITY;
```

```
1 v SELECT ST.CITY, MAX(S.TEMP_F) AS MAX_Temperature,  
2     MIN(S.TEMP_F) AS MIN_Temperature,  
3     AVG(S.TEMP_F) AS AVERAGE_Temperature FROM STATS S  
4 JOIN STATION ST  
5 ON S.ID = ST.ID  
6 GROUP BY CITY;  
7
```

CITY	MAX_TEMPERATURE	MIN_TEMPERATURE	AVERAGE_TEMPERATURE
CARIBOU	65.8	6.7	36.25
DENVER	74.8	27.3	51.05
PHOENIX	91.7	57.4	74.55

Download CSV

3 rows selected.

11. Execute a query to display each city's monthly temperature in Celcius and rainfall in Centimeter.

WORKING-

from stats- TEMP_F {FAHRENHEIT TO CELCIUS} ,
RAIN_I {INCHES TO CENTIMETER} (ID)

```
SELECT ST.CITY, S."MONTH",  
       ((S.TEMP_F - 32) * (5.0/9.0)) AS TEMPERATURE_C,  
       (S.RAIN_I * 2.54) AS RAINFALL_CM  
FROM STATS S  
JOIN STATION ST
```

```
ON S.ID = ST.ID GROUP BY CITY ;
```

```
1 v SELECT ST.CITY, S."MONTH",
2      ((S.TEMP_F - 32) * (5.0/9.0)) AS TEMPERATURE_C,
3      (S.RAIN_I * 2.54) AS RAINFALL_CM
4 FROM STATS S
5 JOIN STATION ST
6 ON S.ID = ST.ID
7
8
9
10
```

[illegible][Download CSV](#)

6 rows selected.

12. Update all rows of table STATS to compensate for faulty rain gauges known to read 0.01 inches low.

```
UPDATE STATS  
SET RAIN_I=RAIN_I + 0.01;  
SELECT * FROM STATS
```

```
1 v UPDATE STATS  
2 SET RAIN_I=RAIN_I + 0.01;  
3  
4 SELECT * FROM STATS  
5  
6
```

6 row(s) updated.

ID	MONTH	TEMP_F	RAIN_I
13	1	57.4	.32
13	7	91.7	5.16
44	1	27.3	.19
44	7	74.8	2.12
66	1	6.7	2.11
66	7	65.8	4.53

13. Update Denver's July temperature reading as 74.9

STEP 1 (TABLE BEFORE UPDATE)

CREATE VIEW JOIN_S_ST AS

SELECT ST.CITY, S."MONTH", S.TEMP_F FROM STATS S

JOIN STATION ST

ON S.ID=ST.ID;

SELECT * FROM JOIN_S_ST;

```

1 v CREATE VIEW JOIN_S_ST AS
2 SELECT ST.CITY, S."MONTH", S.TEMP_F FROM STATS S
3 JOIN STATION ST
4 ON S.ID=ST.ID;
5
6 SELECT * FROM JOIN_S_ST;
7

```

CITY	MONTH	TEMP_F
PHOENIX	1	57.4
PHOENIX	7	91.7
DENVER	1	27.3
DENVER	7	74.8
CARIBOU	1	6.7
CARIBOU	7	65.8

Download CSV

6 rows selected.

STEP 2 (TABLE AFTER UPDATE)

```
UPDATE JOIN_S_ST  
SET TEMP_F = 74.9  
WHERE CITY = 'DENVER' AND "MONTH"=7;
```

```
1 v UPDATE JOIN_S_ST  
2 SET TEMP_F = 74.9  
3 WHERE CITY = 'DENVER' AND "MONTH"=7;  
4  
5 SELECT * FROM JOIN_S_ST;  
6
```

1 row(s) updated.

CITY	MONTH	TEMP_F
PHOENIX	1	57.4
PHOENIX	7	91.7
DENVER	1	27.3
DENVER	7	74.9
CARIBOU	1	6.7
CARIBOU	7	65.8

Download CSV