`.mode column` Left-aligned columns
`.mode csv` Comma-separated values
`.headers on` Turn display of headers on
`.output ?FILENAME?` Send output to FILENAME
`.output stdout` Send output to stdout

1. The following query contains the date, temperature quantity measurement, and the converted temperature readings from Fahrenheit to Celsius rounded to two decimal places.
   The table gives us the temperature in Celsius on corresponding dates, and the temperature of unknown date is removed.

   In this query,

- Three columns are selected
- The temperature reading in Celsius is calculated as `5*(reading-32)/9`, and then rounded to two decimal places `round(5*(reading-32)/9,2)`. The column `reading` is renamed `as Celsius`.
- `JOIN` creates the cross product of table `Survey` and `Visited`, and then use `ON` to filter out combinations of records with `Survey.taken = Visited.id`
- `AND quant = 'temp'` filter `quant` column with only temperature readings
- Remove missing data `Visited.dated IS NOT NULL`, since there is only missing data in `dated` column.

```
sqlite> .mode csv
sqlite> .output Temperature.csv
sqlite> .mode column
sqlite> .headers on
sqlite> SELECT Visited.dated, quant, round(5*(reading-32)/9,2) as Celsius
   ...> FROM Survey JOIN Visited
   ...> ON Survey.taken = Visited.id
   ...> AND quant = 'temp'
   ...> AND Visited.dated IS NOT NULL;
sqlite> .output stdout

dated       quant       Celsius
----------  ----------  ----------
1930-01-07  temp        -29.72
1930-01-12  temp        -32.22
1930-02-26  temp        -28.06
```

2. The following query contains the identifier, Full name (by concatenating personal and family), and the latitude and longitude.
   The table gives us the coordinates and full names of each person.

   In this query,

- Four columns are selected
- Use the concatenation operator `||` to combine `personal` and `family` and rename the concatenating column as `Fullname`.
- `JOIN` creates the cross product of all the four tables, and then use `ON` to filter out combinations of records with `Person.id = Survey.person AND Survey.taken = Visited.id AND Visited.site = Site.name`.
- Then we will actually has a lot of duplicate rows, all the records in each batch have the same value for `Person.id` and `Fullname`, so use `GROUP BY` to group all the records with the same value for the specified field together so that aggregation can process each batch separately.

```
sqlite> .mode csv
sqlite> .output Fullname.csv
sqlite> .mode column
sqlite> .headers on
sqlite> SELECT Person.id, personal || ' ' || family as Fullname, Site.lat, Site.long
   ...> FROM Person JOIN Site JOIN Survey JOIN Visited
   ...> ON Person.id = Survey.person
   ...> AND Survey.taken = Visited.id
   ...> AND Visited.site = Site.name
   ...> GROUP BY Person.id, Fullname;
sqlite> .output stdout

id          Fullname      lat         long
----------  ------------  ----------  ----------
dyer        William Dyer  -49.85      -128.57
lake        Anderson Lak  -48.87      -123.4
pb          Frank Pabodi  -47.15      -126.72
roe         Valentina Ro  -49.85      -128.57
```

3. The following query contains the site, kind of quantity measurement, the minimum reading and maximum reading.
   The table gives us the minimum reading and maximum reading of each kind of quantity measurement taken at each site.

   In this query,

- The `reading` values in `Survey` table are modified by calculating minimum reading and maximum reading.
- Four columns are selected.
- Rename: `min(reading) as MinReading, max(reading) as MaxReading`
- `JOIN` creates the cross product of table `Survey` and `Visited`, and then use `ON` to filter out combinations of records with `Survey.taken = Visited.id`
- All the records in each batch have the same value for `site` and `quant`, so use `GROUP BY` to group all the records with the same value for the specified field together.

```
sqlite> .mode csv
sqlite> .output SiteReadings.csv
sqlite> .mode column
sqlite> .headers on
sqlite> SELECT Visited.site, quant, min(reading) as MinReading, max(reading) as
MaxReading
   ...> FROM Visited JOIN Survey
   ...> ON Visited.id = Survey.taken
   ...> GROUP BY Visited.site, quant;
sqlite> .output stdout

site        quant       MinReading  MaxReading
----------  ----------  ----------  ----------
DR-1        rad         7.8         11.25
DR-1        sal         0.09        0.13
DR-3        rad         2.19        8.41
DR-3        sal         0.05        41.6
DR-3        temp        -26.0       -16.0
MSK-4       rad         1.46        1.46
MSK-4       sal         0.21        22.5
```