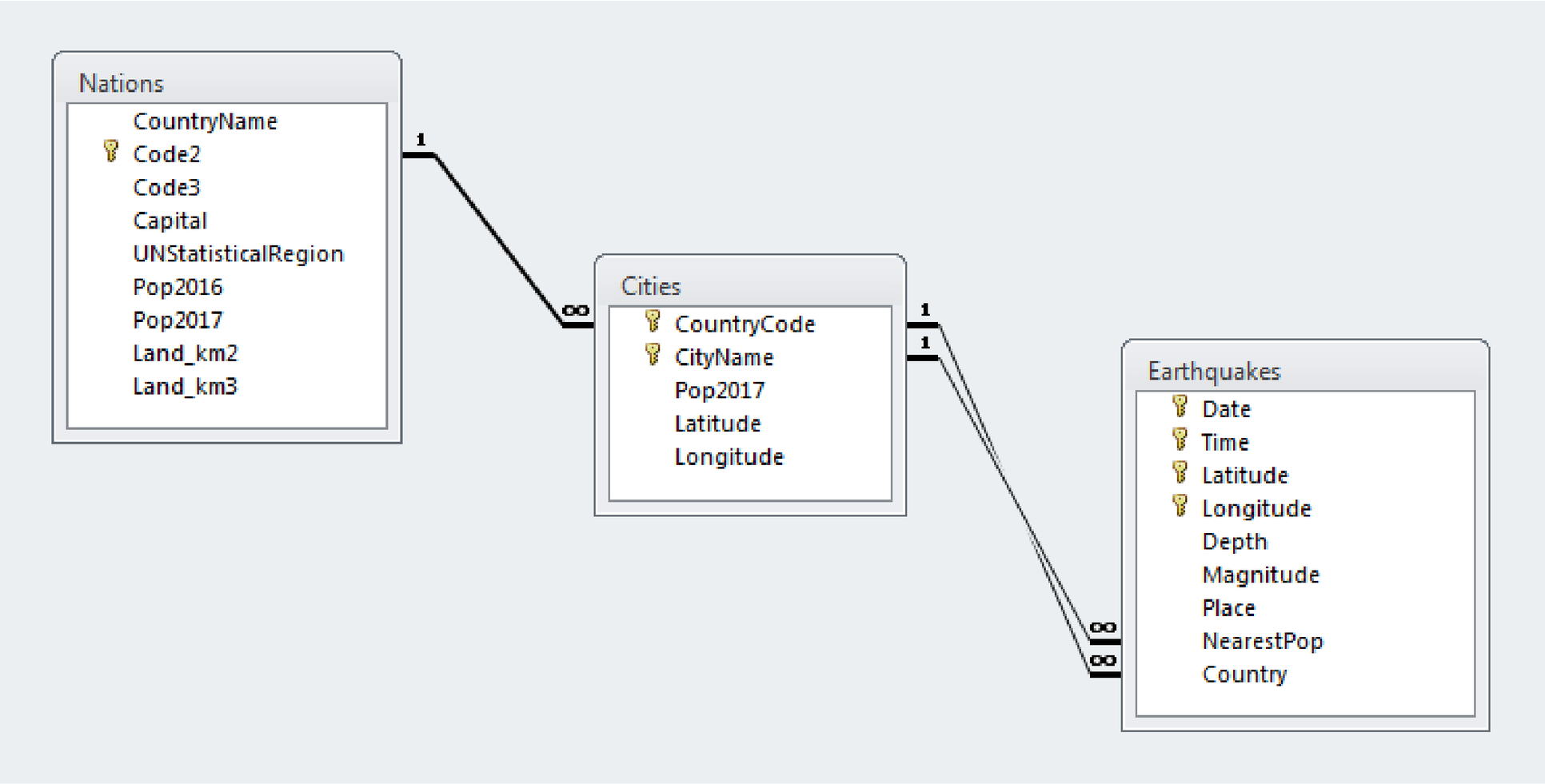
Introduction

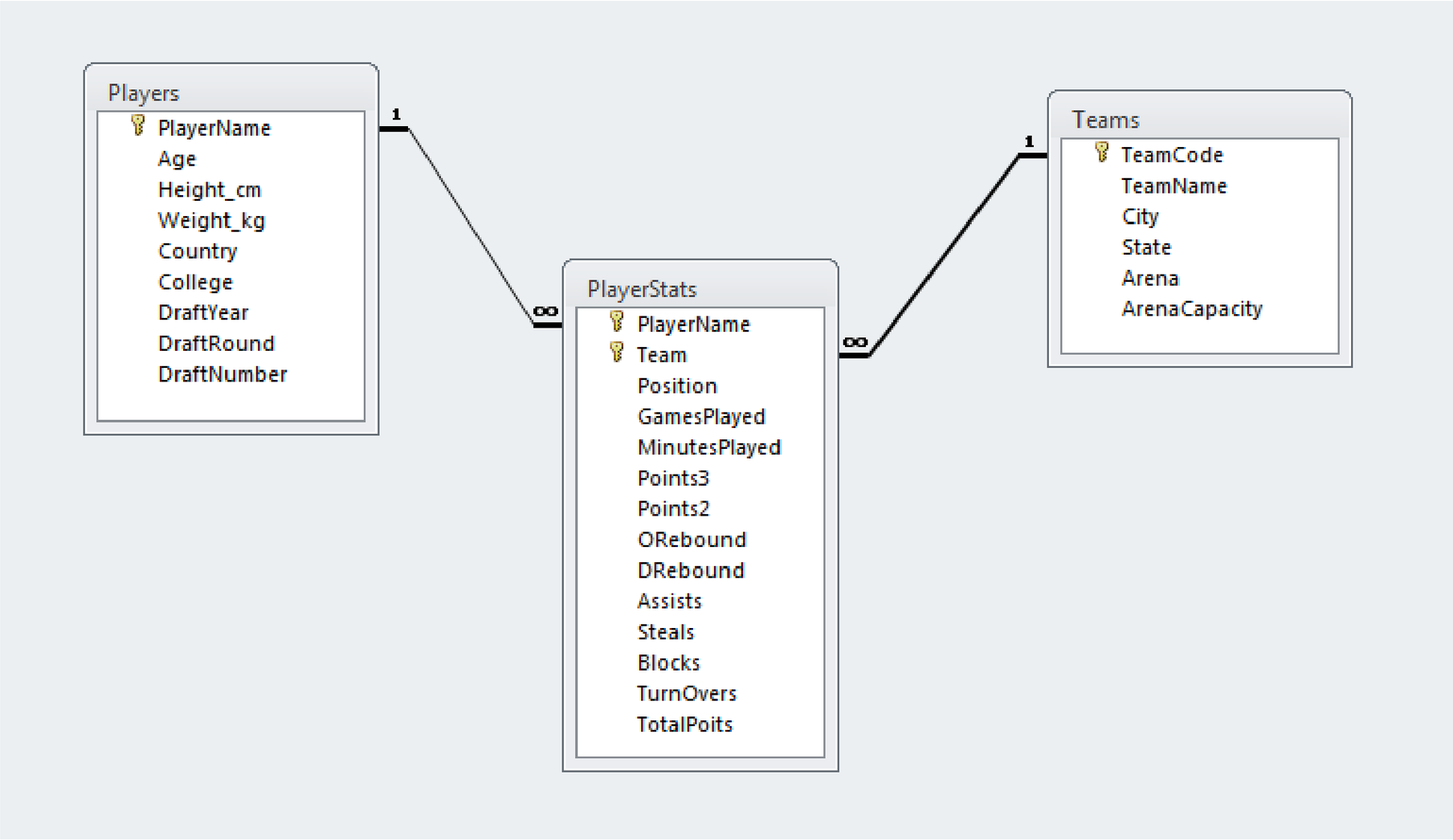
IMPROVING QUERY PERFORMANCE IN SQL SERVER

Dean Smith
Founder, Atamai Analytics

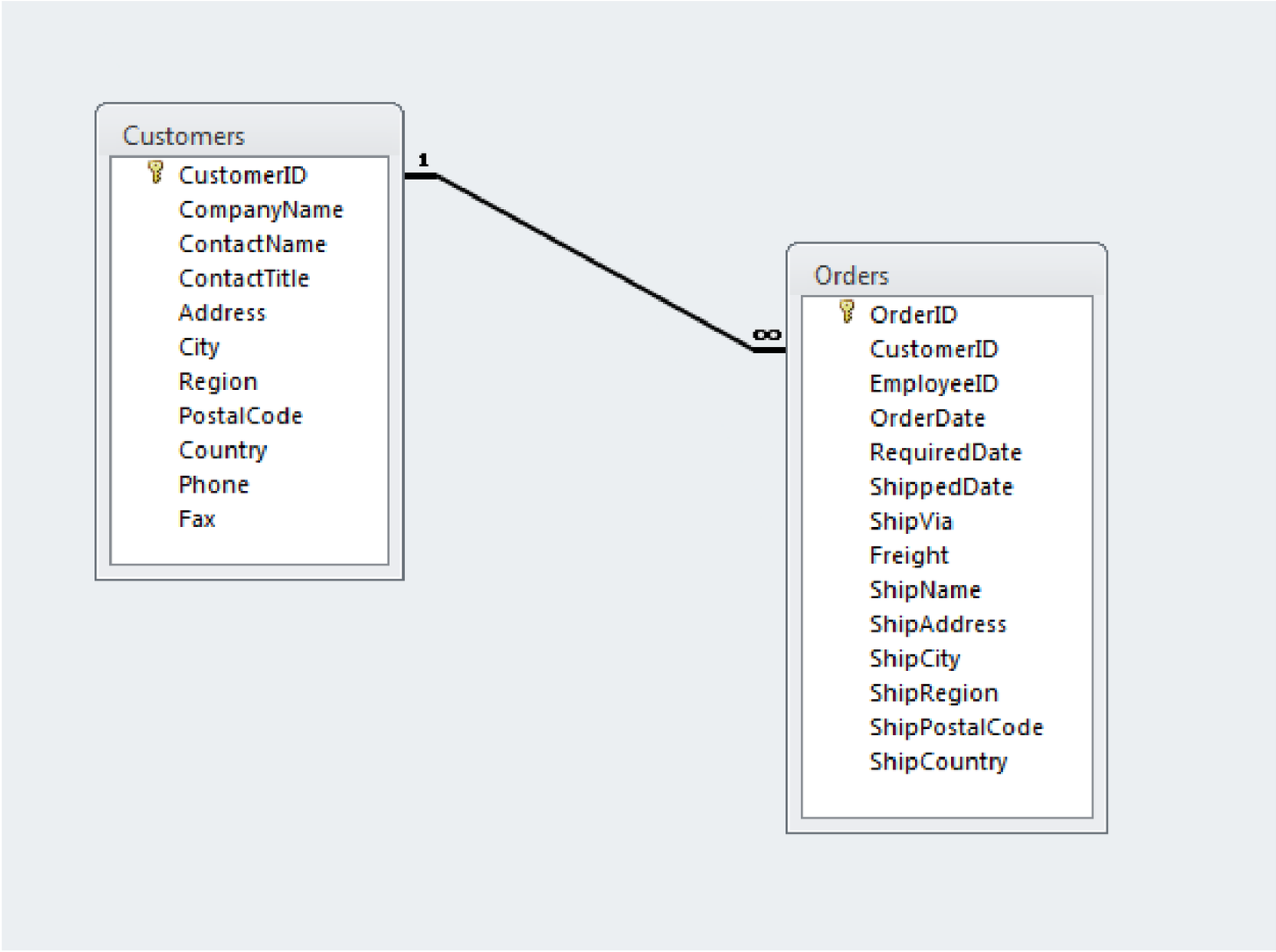
Earthquakes database



NBA Season 2017-2018 database



Customer Orders database



Is this easy to read?

```
Select ps.Team, count(p.PlayerName)
As NonNthAmerPlayers from
 PlayerStats ps inner
join (select PlayerName FROM Players
    WHERE Country <> 'USA' Or Country
 <> 'Canada' )
 p on p.PlayerName = ps.PlayerName
 group BY ps.Team
having Count(p.PlayerName)
>=24 Order by NonNthAmerPlayers desc
```

Team	NonNthAmerPlayers
HOU	24
LAL	24
MEM	24
MIL	24

Suggestions

- Be consistent
- Use UPPER CASE for all SQL syntax
- Create a new line for each major processing syntax: `SELECT`, `FROM`, `WHERE`, etc.
- Indent code:
  - Sub-queries
  - `ON` statements
  - `AND` / `OR` conditions
  - To avoid long single lines of code, for example, several column names
- Complete the query with a semi-colon (;)
- Alias where required, using `AS`

Much better...

From

```
Select ps.Team, count(p.PlayerName)
As NonNthAmerPlayers from
 PlayerStats ps inner
join (select PlayerName FROM Players
    WHERE Country <> 'USA' Or Country
 <> 'Canada' )
 p on p.PlayerName = ps.PlayerName
 group BY ps.Team
having Count(p.PlayerName)
>=24 Order by NonNthAmerPlayers desc
```

To

```
SELECT ps.Team,
    COUNT(p.PlayerName) NonNthAmerPlayers
FROM PlayerStats ps
INNER JOIN
        (SELECT PlayerName
         FROM Players
         WHERE Country <> 'USA'
                OR Country <> 'Canada' ) p
    ON p.PlayerName = ps.PlayerName
GROUP BY ps.Team
HAVING COUNT(p.PlayerName) >=24
ORDER BY NonNthAmerPlayers DESC;
```

Commenting blocks

```
/*
Returns a list of NBA teams with 24 or more non-North
American players on the team roster.
*/
SELECT ps.Team,
  COUNT(p.PlayerName) NonNthAmerPlayers
FROM PlayerStats ps
INNER JOIN
    (SELECT PlayerName
     FROM Players
     WHERE Country <> 'USA'
         OR Country <> 'Canada' ) p
    ON p.PlayerName = ps.PlayerName
GROUP BY ps.Team
HAVING COUNT(p.PlayerName) >=24
ORDER BY NonNthAmerPlayers DESC;
```

Use `/*` and `*/` to comment out a ***block*** of code or text

Commenting blocks

```
/*
Returns a list of NBA teams with 24 or more non-North
American players on the team roster.
*/
SELECT ps.Team,
  COUNT(p.PlayerName) NonNthAmerPlayers
FROM PlayerStats ps
INNER JOIN
    (SELECT PlayerName
     FROM Players
     WHERE Country <> 'USA'
         OR Country <> 'Canada' ) p
    ON p.PlayerName = ps.PlayerName
GROUP BY ps.Team
HAVING COUNT(p.PlayerName) >=24
ORDER BY NonNthAmerPlayers DESC;
```

Use `/*` and `*/` to comment out a ***block*** of code or text

Team	NonNthAmerPlayers
HOU	24
LAL	24
MEM	24
MIL	24

Commenting lines

Use -- to comment out a single ***line*** of code or text

```
SELECT ps.Team,
  COUNT(p.PlayerName) NonNthAmerPlayers
FROM PlayerStats ps

INNER JOIN
    (SELECT PlayerName
     FROM Players
     WHERE Country <> 'USA'
        OR Country <> 'Canada' ) p
    ON p.PlayerName = ps.PlayerName
GROUP BY ps.Team
HAVING COUNT(p.PlayerName) >=24

ORDER BY NonNthAmerPlayers DESC;
```

Commenting lines

Use -- to comment out a single ***line*** of code or text

```
SELECT ps.Team,
  COUNT(p.PlayerName) NonNthAmerPlayers -- Count of players
FROM PlayerStats ps

INNER JOIN
    (SELECT PlayerName
     FROM Players
     WHERE Country <> 'USA'
        OR Country <> 'Canada' ) p -- Indented qub-suery
    ON p.PlayerName = ps.PlayerName
GROUP BY ps.Team
HAVING COUNT(p.PlayerName) >=24

ORDER BY NonNthAmerPlayers DESC;
```

- Comment indicating that the new column is a count of players
- Comment indicating that the sub-query is indented

Commenting lines

Use -- to comment out a single ***line*** of code or text

```
SELECT ps.Team,
  COUNT(p.PlayerName) NonNthAmerPlayers -- Count of players
FROM PlayerStats ps
-- Inner join starts here
INNER JOIN
    (SELECT PlayerName
     FROM Players
     WHERE Country <> 'USA'
        OR Country <> 'Canada' ) p -- Indented qub-suery
    ON p.PlayerName = ps.PlayerName
GROUP BY ps.Team
HAVING COUNT(p.PlayerName) >=24
-- Remove the ORDER BY, it is not required
ORDER BY NonNthAmerPlayers DESC;
```

- Comment marking a break before the `INNER JOIN`
- Comment about the requirement of `ORDER BY`

Commenting lines

Use -- to comment out a single ***line*** of code or text

```
SELECT ps.Team,
  COUNT(p.PlayerName) NonNthAmerPlayers -- Count of players
FROM PlayerStats ps
-- Inner join starts here
INNER JOIN
    (SELECT PlayerName
     FROM Players
     WHERE Country <> 'USA'
        OR Country <> 'Canada' ) p -- Indented qub-suery
    ON p.PlayerName = ps.PlayerName
GROUP BY ps.Team
HAVING COUNT(p.PlayerName) >=24;
-- Remove the ORDER BY, it is not required
-- ORDER BY NonNthAmerPlayers DESC
```

- Commented out `ORDER BY` statement

Let's practice

IMPROVING QUERY PERFORMANCE IN SQL SERVER

Aliasing

IMPROVING QUERY PERFORMANCE IN SQL SERVER

Dean Smith
Founder, Atamai Analytics

What is aliasing?

- Used in queries to identify:
  - Tables
  - Columns
  - Sub-queries
- Temporary, only applied when the query is run
- Makes the query easier to read
- May be required

Why use aliasing?

- Avoid repetitive use of long table or column names
- Easily identify joined tables and associated columns
- Identify new columns
- Identify sub-queries
- Avoid ambiguity when columns from joined tables share the same name
- Rename columns

Joined tables - ambiguous column name

```
SELECT CountryName,
       Code2,
       Capital,
       Pop2017
FROM Nations
INNER JOIN Cities
  ON Capital = CityName;
```

```
-----------------------------------------------
-- ERROR, Pop2017 column is in
both the Nations and Cities tables

Ambiguous column name 'Pop2017'.
```

Joined tables - aliasing table names

```
-- Alias tables; Nations as n and Cities as c
SELECT n.CountryName,
       n.Code2,
       n.Capital,
       c.Pop2017 -- City population
FROM Nations AS n
INNER JOIN Cities AS c
  ON n.Capital = c.CityName;
```

CountryName	Code2	Capital	Pop2017
United Kingdom	GB	London	346774
Canada	CA	Ottawa	874433
France	FR	Paris	10437
Reunion	RE	Saint-Denis	1067
...	...	...	...

Renamed columns

```
-- Alias columns;
SELECT n.CountryName AS Country,
       n.Code2 AS CountryCode,
       n.Capital,
       c.Pop2017 AS Population
FROM Nations AS n
INNER JOIN Cities AS c
  ON n.Capital = c.CityName;
```

Country	CountryCode	Capital	Populatio
United Kingdom	GB	London	346774
Canada	CA	Ottawa	874433
France	FR	Paris	10437
Reunion	RE	Saint-Denis	1067
...	...	...	...

New columns

```
-- New colunm aliased as MaxMagnitude
SELECT Country,
       NearestPop AS City,
       MAX(Magnitude) AS MaxMagnitude
FROM Earthquakes
GROUP BY Country, NearestPop;
```

Country	City	MaxMagnitude
PE	Acar	7.1
US	Aguadilla	7.7
MX	Aguililla	7.2
PW	Airai	7.8
PG	Aitape	7.6
...	...	...

Sub-queries

```
SELECT n.CountryName AS Country,
       n.Capital,
       e.MaxMagnitude
FROM Nations n
INNER JOIN
        (SELECT Country, NearestPop AS City
                ,MAX(Magnitude) AS MaxMagnitude
        FROM Earthquakes
        GROUP BY Country, NearestPop) e
              -- Sub-query aliased as e
    ON n.Code2 = e.Country AND n.Capital = e.City;
```

Country	Capital	MaxMagnitude
Fiji	Suva	7.9
Guam	Hagatna	7.8
Peru	Lima	7.6
Turkmenistan	Ashgabat	7.3
...	...	...

Let's practice

IMPROVING QUERY PERFORMANCE IN SQL SERVER

Query order

IMPROVING QUERY PERFORMANCE IN SQL SERVER

Dean Smith
Founder, Atamai Analytics

Big earthquakes query

```
SELECT Country, Place, Magnitude
FROM Earthquakes
WHERE Magnitude >= 9
ORDER BY Magnitude DESC;
```

Country	Place	Magnitude
CL	Bio-Bio; Chile	9.5
US	Southern Alaska	9.2
ID	off the west coast of northern Sumatra	9.1
JP	near the east coast of Honshu; Japan	9.1
...	...	...

Syntax order

```
                                -- Syntax Order
SELECT Country, Place, Magnitude -- 1. SELECT
FROM Earthquakes
WHERE Magnitude >= 9
ORDER BY Magnitude DESC;
```

Syntax order

```
                                -- Syntax Order
SELECT Country, Place, Magnitude -- 1. SELECT
FROM Earthquakes                 -- 2. FROM
WHERE Magnitude >= 9
ORDER BY Magnitude DESC;
```

Syntax order

```
                                -- Syntax Order
SELECT Country, Place, Magnitude -- 1. SELECT
FROM Earthquakes                 -- 2. FROM
WHERE Magnitude >= 9             -- 3. WHERE
ORDER BY Magnitude DESC;
```

Syntax order

```
                                -- Syntax Order
SELECT Country, Place, Magnitude -- 1. SELECT
FROM Earthquakes                 -- 2. FROM
WHERE Magnitude >= 9             -- 3. WHERE
ORDER BY Magnitude DESC;         -- 4. ORDER BY
```

Processing order

```
                                -- Syntax Order | Processing Order
SELECT Country, Place, Magnitude -- 1. SELECT
FROM Earthquakes                -- 2. FROM         1. FROM
WHERE Magnitude >= 9            -- 3. WHERE
ORDER BY Magnitude DESC;        -- 4. ORDER BY
```

Processing order

```
                                -- Syntax Order | Processing Order
SELECT Country, Place, Magnitude -- 1. SELECT
FROM Earthquakes                -- 2. FROM         1. FROM
WHERE Magnitude >= 9            -- 3. WHERE        2. WHERE
ORDER BY Magnitude DESC;        -- 4. ORDER BY
```

Processing order

```
                                -- Syntax Order | Processing Order
SELECT Country, Place, Magnitude -- 1. SELECT      3. SELECT
FROM Earthquakes                -- 2. FROM        1. FROM
WHERE Magnitude >= 9            -- 3. WHERE       2. WHERE
ORDER BY Magnitude DESC;        -- 4. ORDER BY
```

Processing order

```
                                -- Syntax Order | Processing Order
SELECT Country, Place, Magnitude -- 1. SELECT      3. SELECT
FROM Earthquakes                -- 2. FROM        1. FROM
WHERE Magnitude >= 9            -- 3. WHERE       2. WHERE
ORDER BY Magnitude DESC;        -- 4. ORDER BY    4. ORDER BY
```

Processing errors

```
SELECT Country,
       PlaceName,
       Magnitude
FROM LargeEarthquakes
WHERE Strength >= 9
ORDER BY Magnitud DESC;
```

Processing FROM

```
                        -- Processing Order
SELECT Country,
       PlaceName,
       Magnitude
FROM LargeEarthquakes   -- 1. FROM - table LargeEarthquakes does not exist
WHERE Strength >= 9
ORDER BY Magnitud DESC;
```

```
-------------------------------------------------------------------------------
-- ERROR
Invalid object name 'LargeEarthquakes'.
```

Processing WHERE

```
                        -- Processing Order
SELECT Country,
       PlaceName,
       Magnitude
FROM Earthquakes
WHERE Strength >= 9      -- 2. WHERE - column Strength does not exist
ORDER BY Magnitud DESC;
```

```
-------------------------------------------------------------------------
-- ERROR
Invalid column name 'Strength'.
```

Processing SELECT

```
                        -- Processing Order
SELECT Country,
       PlaceName,       -- 3. SELECT - column PlaceName does not exist
       Magnitude
FROM Earthquakes
WHERE Magnitude >= 9
ORDER BY Magnitud DESC;
```

```
-------------------------------------------------------------------------
-- ERROR
Invalid column name 'PlaceName'.
```

Processing ORDER BY

```
                        -- Processing Order
SELECT Country,
       Place,
       Magnitude
FROM Earthquakes
WHERE Magnitude >= 9
ORDER BY Magnitud DESC;-- 4. ORDER BY - column misspelling
```

```
-----------------------------------------------------------

-- ERROR
Invalid column name 'Magnitud'.
```

Error free

```
SELECT Country,
       Place,
       Magnitude
FROM Earthquakes
WHERE Magnitude >= 9
ORDER BY Magnitude DESC;
```

Country	Place	Magnitude
CL	Bio-Bio; Chile	9.5
US	Southern Alaska	9.2
ID	off the west coast of northern Sumatra	9.1
JP	near the east coast of Honshu; Japan	9.1
...	...	...

Logical processing order

1. FROM
2. ON
3. JOIN
4. WHERE
5. GROUP BY
6. HAVING

7. SELECT

8. DISTINCT
9. ORDER BY
10. TOP

Let's practice

IMPROVING QUERY PERFORMANCE IN SQL SERVER