Planning the query

REPORTING IN SQL

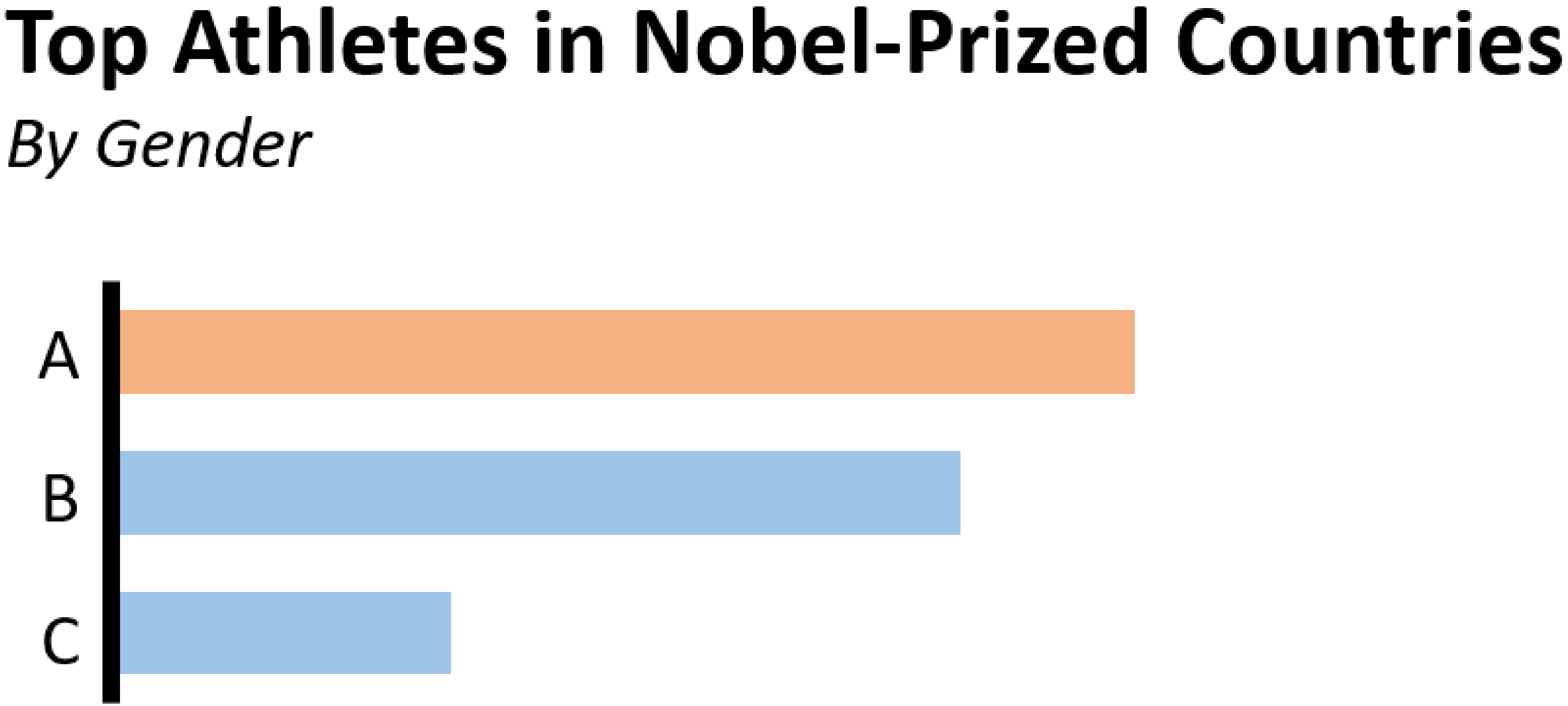
SQL

Tyler Pernes

Learning & Development Consultant

datacamp

Chapter goal



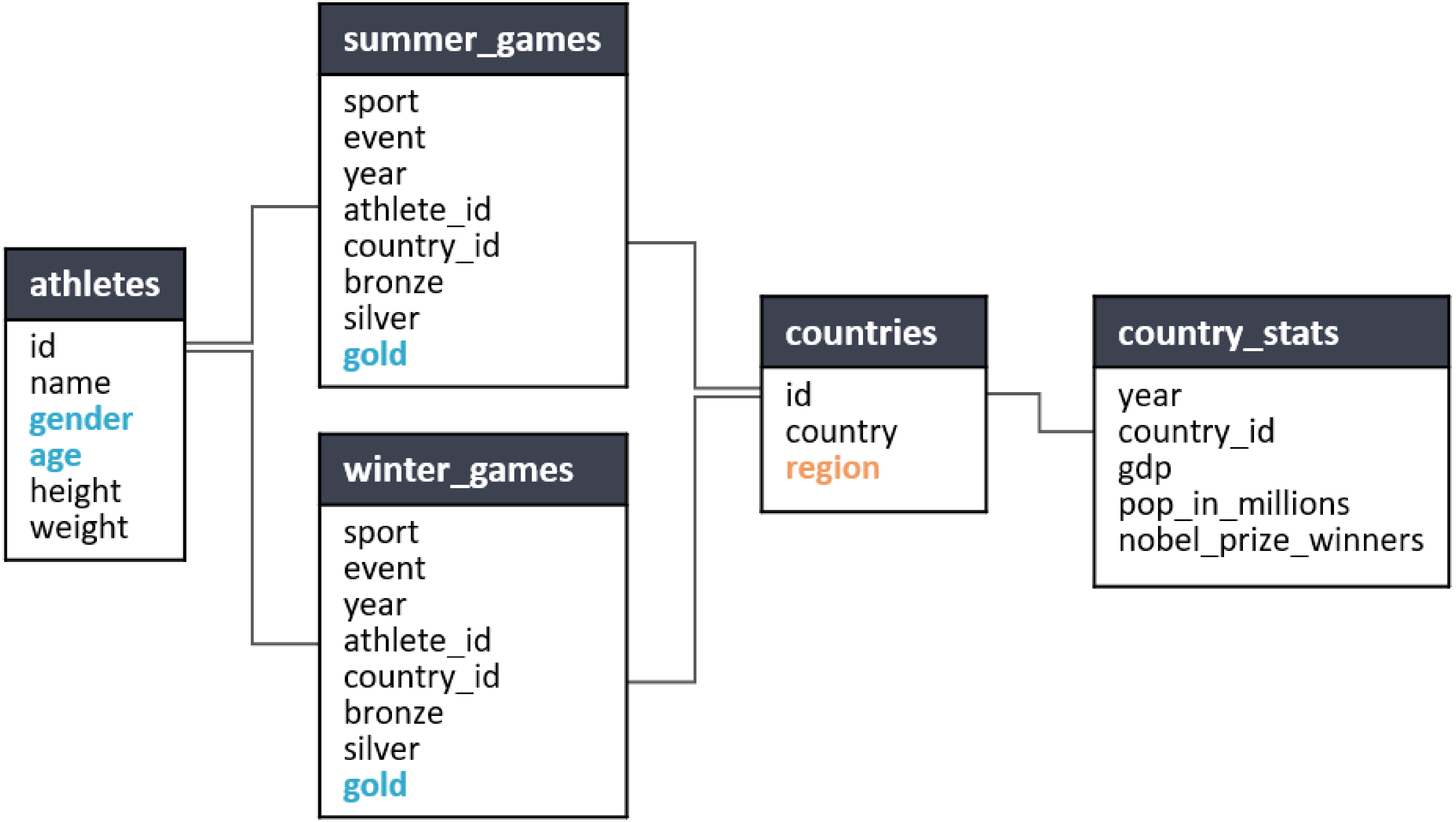
Questions to ask

- What **tables** do we need to pull from?
- How should we **combine** the tables?
- What **fields** do we need to create?
- What **filters** need to be included?
- Any **ordering** or **limiting** needed?

Scenario

```
Gold Medals by Demographic Group
(Western European Countries Only)
+----------+--------------------+-------+
| season    |  demographic_group | golds |
|----------|--------------------|-------|
| Winter    | Male Age 26+       | 13    |
| Winter    | Female Age 26+     | 8     |
| Summer    | Male Age 13-25     | 7     |
| Summer    | Female Age 13-25   | 6     |
| Winter    | Male Age 13-25     | 4     |
| Summer    | Male Age 26+       | 4     |
| Winter    | Female Age 13-25   | 4     |
| Summer    | Female Age 26+     | 2     |
+----------+--------------------+-------+
```

1 - What tables do we need to pull from?



summer_games
sport
event
year
athlete_id
country_id
bronze
silver
gold

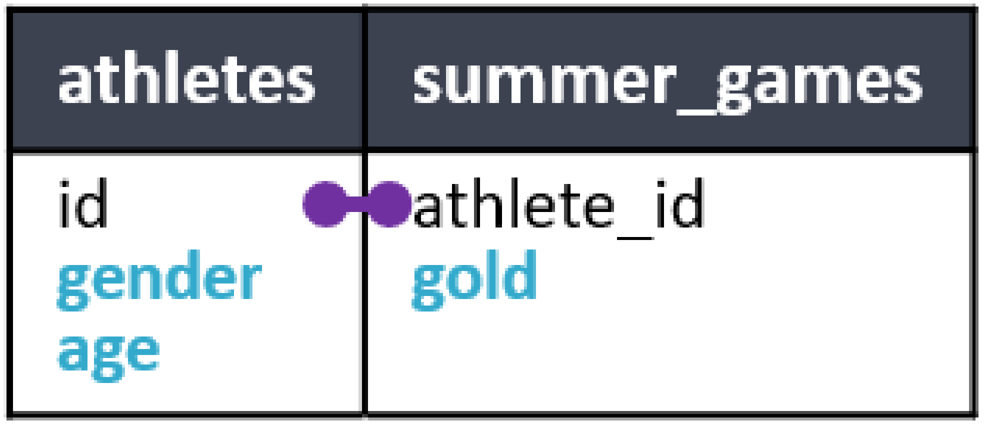
athletes
id
name
gender
age
height
weight

winter_games
sport
event
year
athlete_id
country_id
bronze
silver
gold

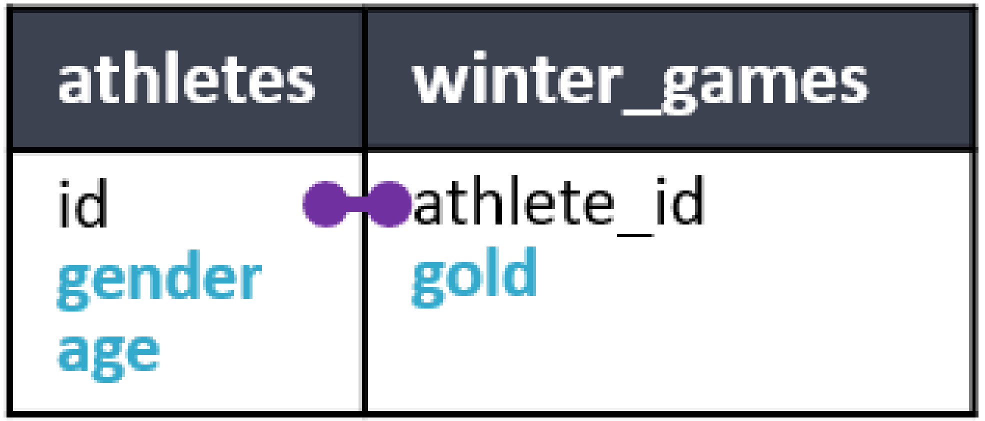
countries
id
country
region

country_stats
year
country_id
gdp
pop_in_millions
nobel_prize_winners

2 - How should we combine the tables?

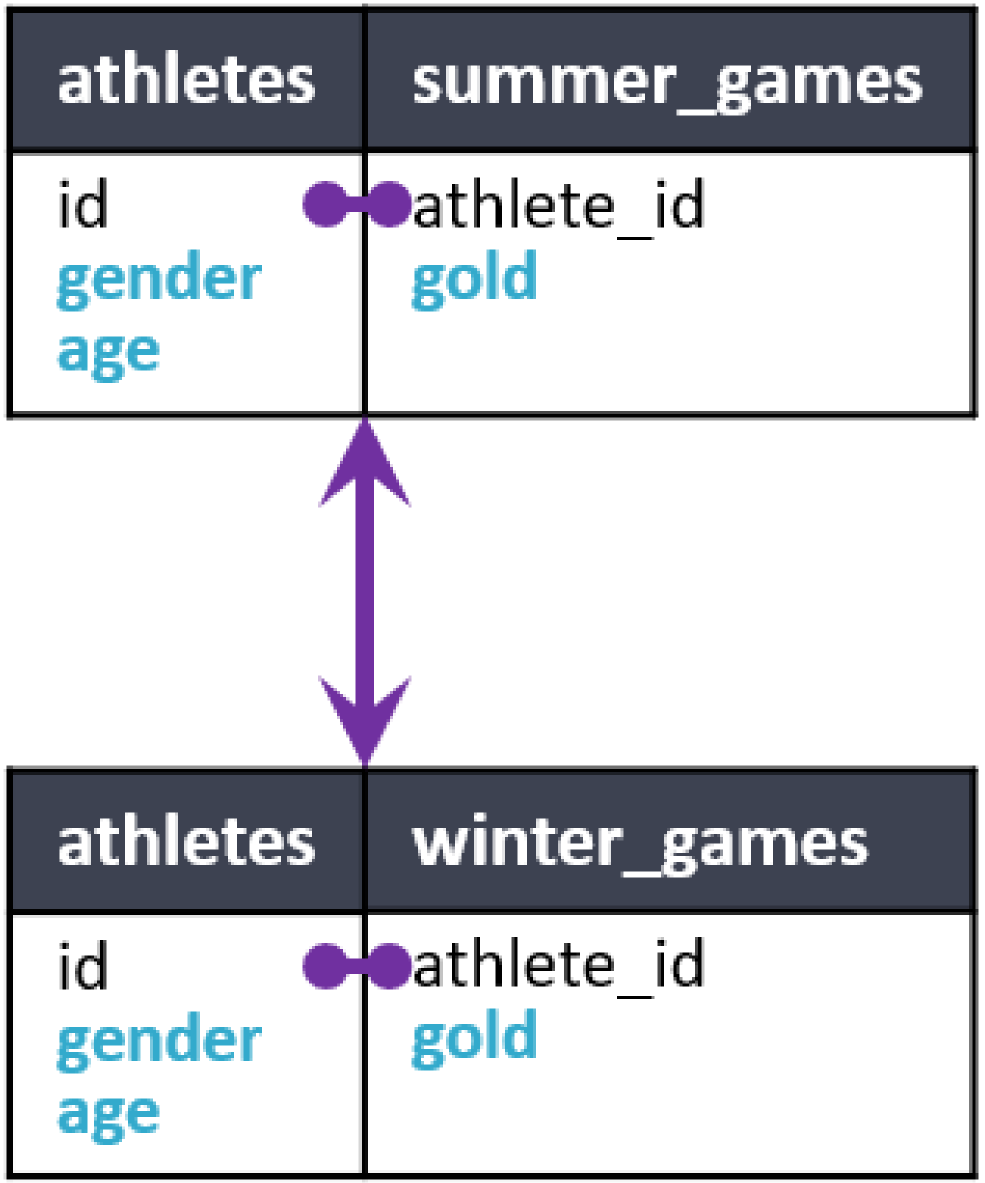


athletes	summer_games
id gender age	athlete_id gold



athletes	winter_games
id gender age	athlete_id gold

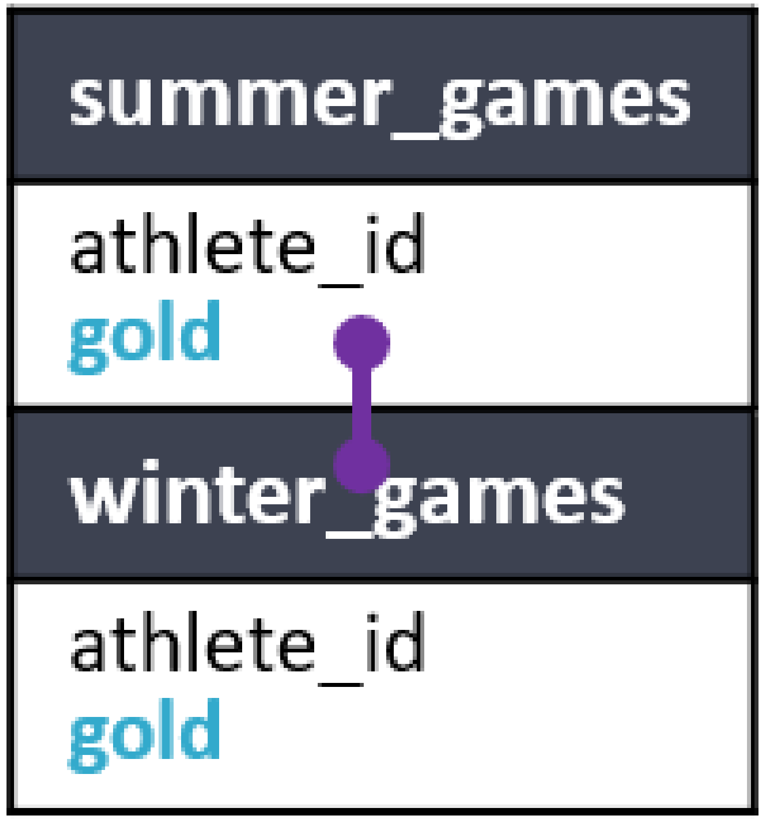
2 - How should we combine the tables?



athletes	summer_games
id	athlete_id
gender	gold
age	

athletes	winter_games
id	athlete_id
gender	gold
age	

2 - How should we combine the tables?



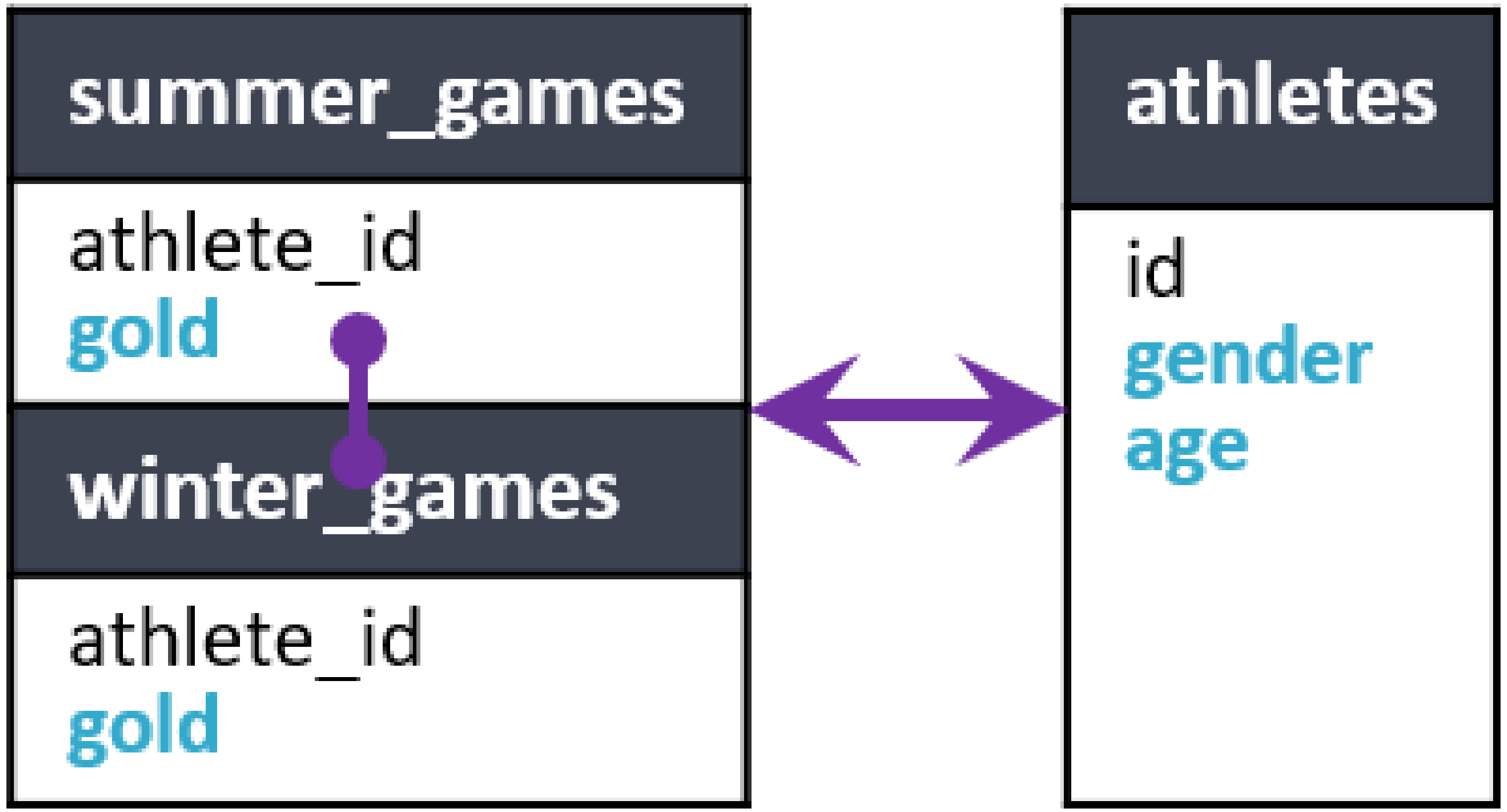
summer_games

athlete_id
gold

winter_games

athlete_id
gold

2 - How should we combine the tables?



3 - What fields do we need to create?

```
+----------+--------------------+-------+
| season   |  demographic_group | golds |
|----------|--------------------|-------|
| Winter   | Male Age 26+       | 13    |
| Winter   | Female Age 26+     | 8     |
| Summer   | Male Age 13-25     | 7     |
| Summer   | Female Age 13-25   | 6     |
| Winter   | Male Age 13-25     | 4     |
| Summer   | Male Age 26+       | 4     |
+----------+--------------------+-------+
```

- **`season`** - `static string`
- **`demographic_group`** - `conditional`
- **`golds`** - `SUM()`

4 - What filters need to be included?

```
Gold Medals by Demographic Group
(Western European Countries Only)
+----------+--------------------+-------+
| season   |  demographic_group | golds |
|----------|--------------------|-------|
| Winter   | Male Age 26+       | 13    |
| Winter   | Female Age 26+     | 8     |
| Summer   | Male Age 13-25     | 7     |
| Summer   | Female Age 13-25   | 6     |
| Winter   | Male Age 13-25     | 4     |
| Summer   | Male Age 26+       | 4     |
+----------+--------------------+-------+
```

- `WHERE` or `HAVING` ?
- Filter on dimension = `WHERE`

5 - Any ordering or limiting needed?

```
Gold Medals by Demographic Group
(Western European Countries Only)
+----------+--------------------+-------+
| season   |  demographic_group | golds |
|----------|--------------------|-------|
| Winter   | Male Age 26+       | 13    |
| Winter   | Female Age 26+     | 8     |
| Summer   | Male Age 13-25     | 7     |
| Summer   | Female Age 13-25   | 6     |
| Winter   | Male Age 13-25     | 4     |
| Summer   | Male Age 26+       | 4     |
+----------+--------------------+-------+
```

- No `LIMIT` needed
- Sort by `golds` in descending order

Let's practice!

Combining tables

REPORTING IN SQL

SQL

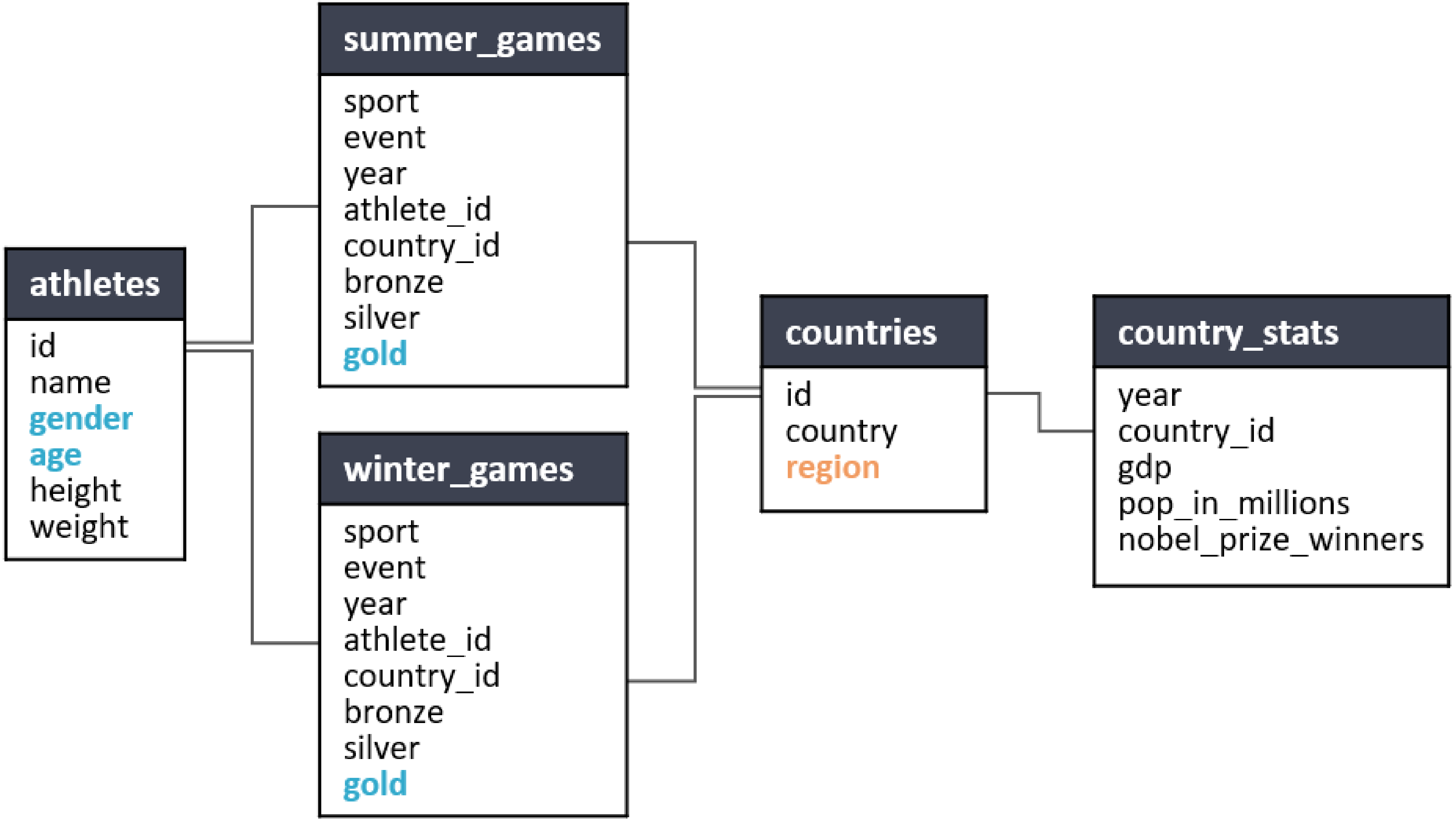
Tyler Pernes

Learning & Development Consultant

Goal report

```
Gold Medals by Demographic Group
(Western European Countries Only)
+----------+--------------------+-------+
| season   |  demographic_group | golds |
|----------+--------------------+-------|
| Winter   | Male Age 26+       | 13    |
| Winter   | Female Age 26+     | 8     |
| Summer   | Male Age 13-25     | 7     |
| Summer   | Female Age 13-25   | 6     |
| Winter   | Male Age 13-25     | 4     |
| Summer   | Male Age 26+       | 4     |
| Winter   | Female Age 13-25   | 4     |
| Summer   | Female Age 26+     | 2     |
+----------+--------------------+-------+
```

Relevant tables



athletes
id
name
gender
age
height
weight

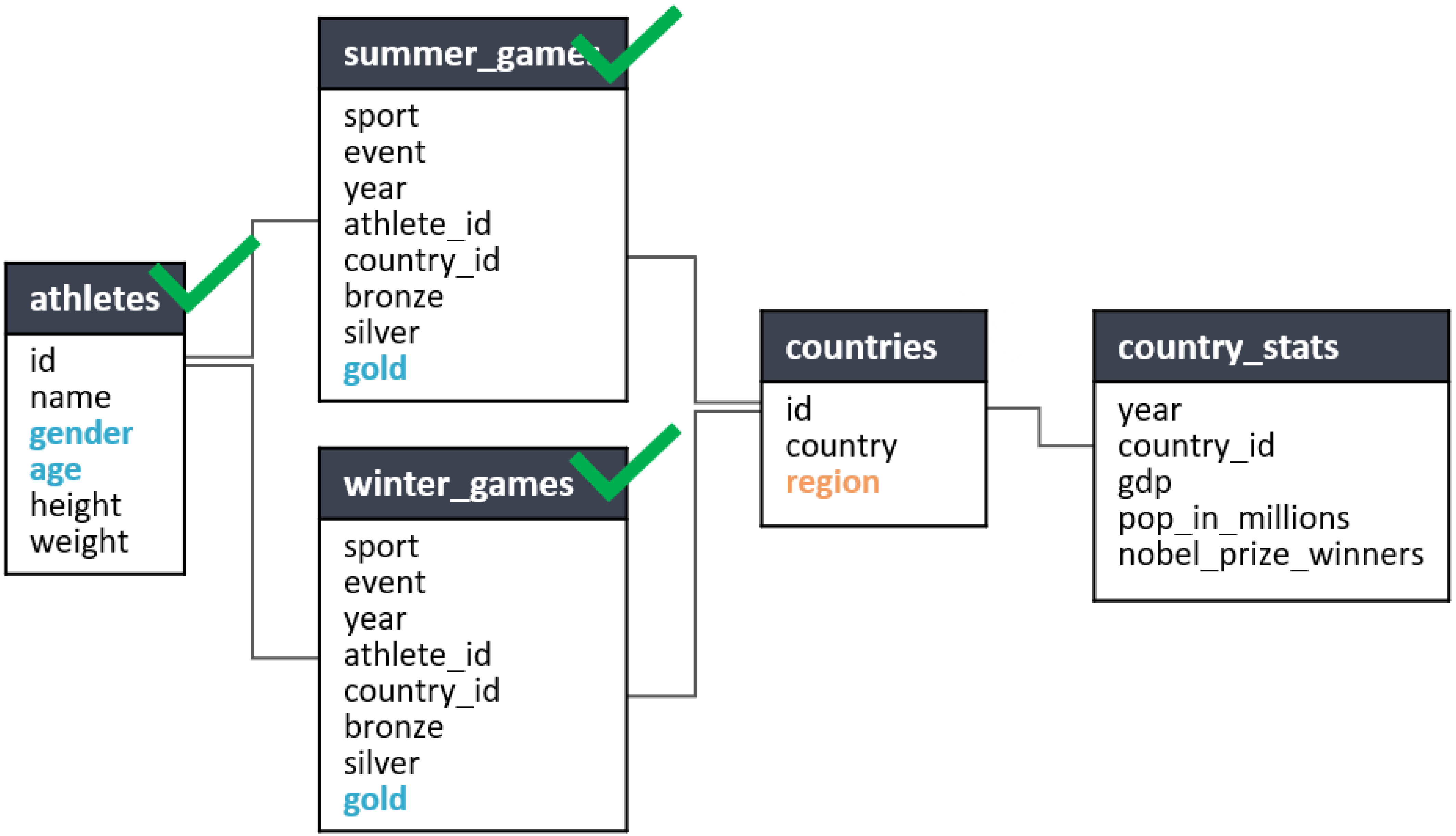
summer_games
sport
event
year
athlete_id
country_id
bronze
silver
gold

winter_games
sport
event
year
athlete_id
country_id
bronze
silver
gold

countries
id
country
region

country_stats
year
country_id
gdp
pop_in_millions
nobel_prize_winners

Relevant tables



summer_games ✓

sport
event
year
athlete_id
country_id
bronze
silver
gold

athletes ✓

id
name
gender
age
height
weight

winter_games ✓

sport
event
year
athlete_id
country_id
bronze
silver
gold

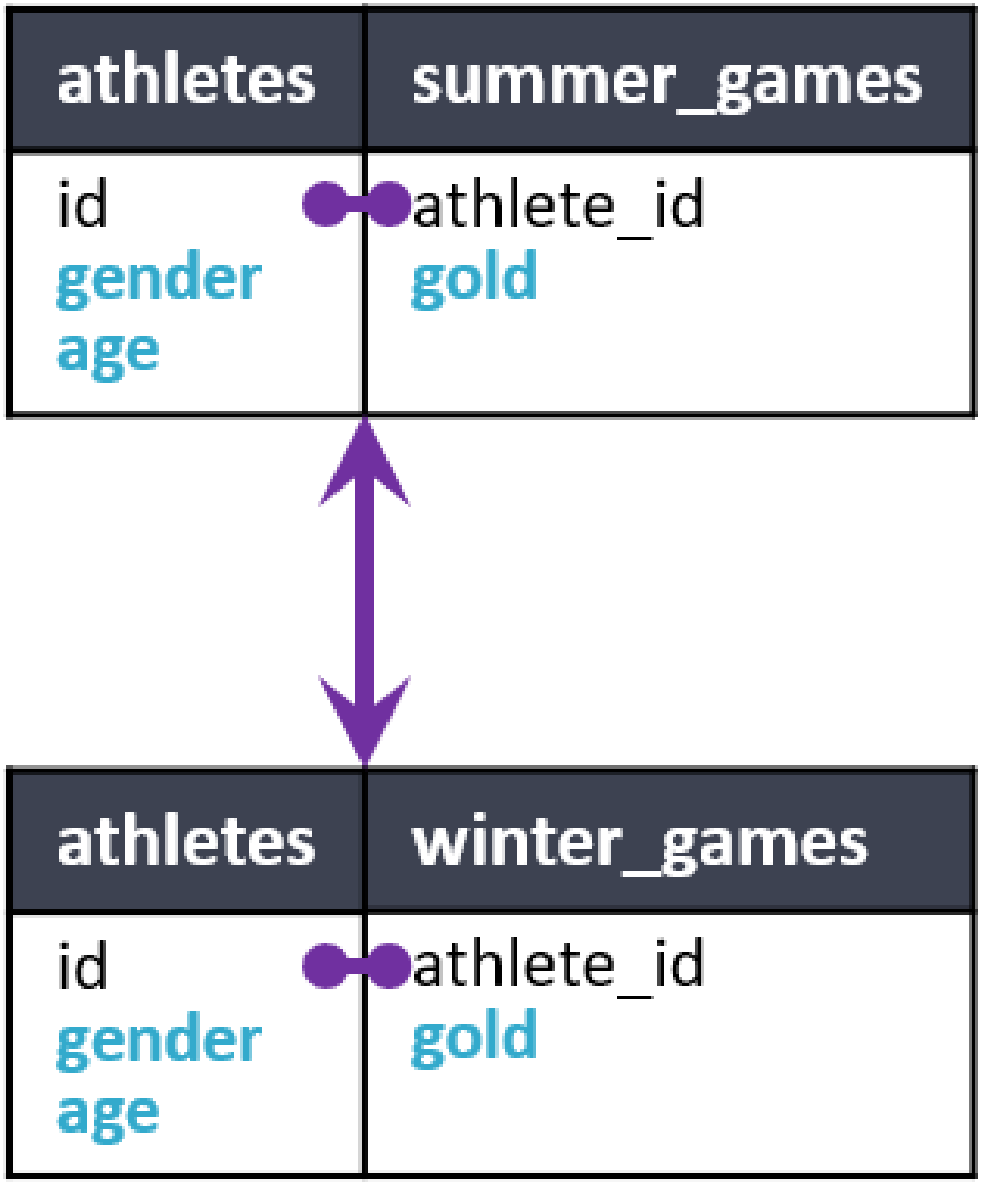
countries

id
country
region

country_stats

year
country_id
gdp
pop_in_millions
nobel_prize_winners

Option A: JOIN first, UNION second



athletes	summer_games
id gender age	athlete_id gold

athletes	winter_games
id gender age	athlete_id gold

Option A: JOIN first, UNION second

Step 1: Setup top query with `JOIN`

```
SELECT
    athlete_id,
    gender,
    age,
    gold
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;
```

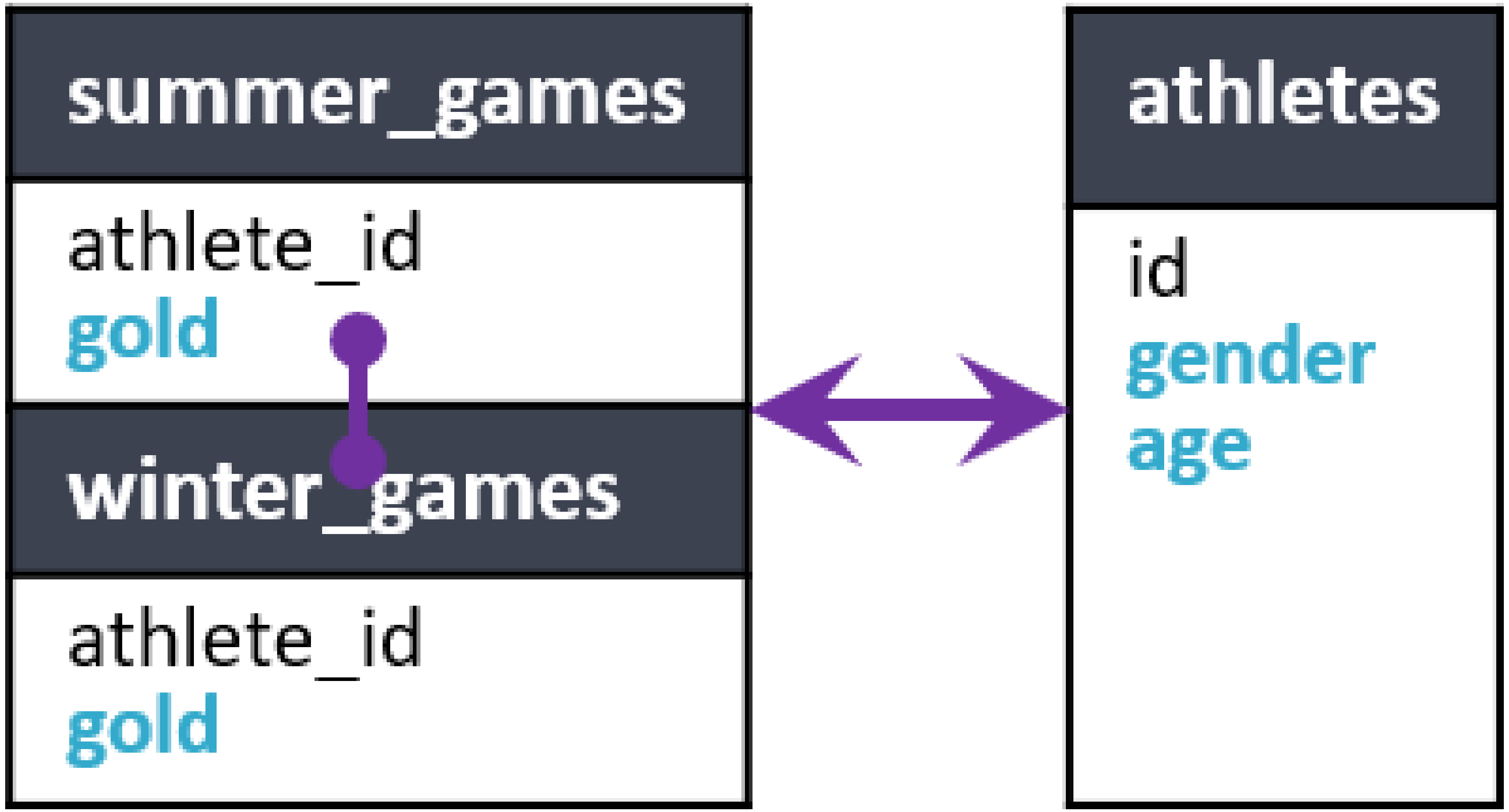
```
Query ran successfully!
```

Option A: JOIN first, UNION second

Step 2: Setup bottom query + `UNION` the two

```
SELECT
    athlete_id,
    gender,
    age,
    gold
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
UNION ALL
SELECT
    athlete_id,
    gender,
    age,
    gold
FROM winter_games AS wg
JOIN athletes AS a
ON wg.athlete_id = a.id;
```

Option B: UNION first, JOIN second



Option B: UNION first, JOIN second

Step 1: Create initial `UNION`

```
SELECT
    athlete_id,
    gold
FROM summer_games AS sg
UNION
SELECT
    athlete_id,
    gold
FROM winter_games AS wg;
```

Option B: UNION first, JOIN second

Step 2: Convert to subquery + `JOIN`

```
SELECT
    athlete_id,
    gender,
    age,
    gold
FROM
    (SELECT
         athlete_id,
         gold
    FROM summer_games AS sg
    UNION ALL
    SELECT athlete_id, gold
    FROM winter_games AS wg) AS g
JOIN athletes AS a
ON g.athlete_id = a.id;
```

Comparison

Option A

```sql
SELECT
    athlete_id,
    gender,
    age,
    gold
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
UNION ALL
SELECT
    athlete_id,
    gender,
    age,
    gold
FROM winter_games AS wg
JOIN athletes AS a
ON wg.athlete_id = a.id;
```

Option B

```sql
SELECT
    athlete_id,
    gender,
    age,
    gold
FROM
    (SELECT
        athlete_id,
        gold
    FROM summer_games AS sg
    UNION ALL
    SELECT athlete_id, gold
    FROM winter_games AS wg) AS g
JOIN athletes AS a
ON g.athlete_id = a.id;
```

Key takeaways

- Several ways to create the same report
- Step-by-step = easier to troubleshoot

Query time!

REPORTING IN SQL

Creating custom fields

REPORTING IN SQL

SQL

Tyler Pernes
Learning & Development Consultant

datacamp

Goal report

```
+----------+--------------------+-------+
| season   |  demographic_group | golds |
|----------+--------------------+-------|
| Winter   | Male Age 26+       | 13    |
| Winter   | Female Age 26+     | 8     |
| Summer   | Male Age 13-25     | 7     |
+----------+--------------------+-------+
```

```
SELECT athlete_id, gender, age, gold
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
UNION ALL
SELECT athlete_id, gender, age, gold
FROM winter_games AS wg
JOIN athletes AS a
ON wg.athlete_id = a.id;
```

Preparation

Step 1: Comment out bottom half

```
SELECT athlete_id, gender, age, gold
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;

/*UNION ALL
SELECT athlete_id, gender, age, gold
FROM winter_games AS wg
JOIN athletes AS a
ON wg.athlete_id = a.id;*/
```

Preparation

Step 2: Add new field placeholders

```
SELECT
    --___ AS season,
    --___ AS demographic_group,
    --___ AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;

/*UNION ALL
SELECT athlete_id, gender, age, gold
FROM winter_games AS wg
JOIN athletes AS a
ON wg.athlete_id = a.id;*/
```

Field 1: seasons

```sql
SELECT
    'Summer' AS season,
    --___ AS demographic_group,
    --___ AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;
```

Field 2: golds

```
SELECT
    'Summer' AS season,
    --___ AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;
```

```
+---------+-------+
| season  | golds |
|---------+-------|
| Summer  | 159   |
+---------+-------+
```

Field 3: demographic_group

```
+--------+-------+--------------------+
| gender | age   | demographic_group  |
|--------+-------+--------------------|
| M      | 18    | Male Age 13-25     |
| M      | 31    | Male Age 26+       |
| F      | 22    | Female Age 13-25   |
| F      | 26    | Female Age 26+     |
+--------+-------+--------------------+
```

CASE statement

```
CASE WHEN {condition_1} THEN {output_1}
WHEN {condition_2} THEN {output_2}
ELSE {output_3}
END
```

Field 3: demographic_group

```sql
SELECT
    'Summer' AS season,
    CASE WHEN ___ THEN 'Male Age 13-25'
    WHEN ___ THEN 'Male Age 26+'
    WHEN ___ THEN 'Female Age 13-25'
    WHEN ___ THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;
```

Field 3: demographic_group

```
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN ___ THEN 'Male Age 26+'
    WHEN ___ THEN 'Female Age 13-25'
    WHEN ___ THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;
```

Field 3: demographic_group

```sql
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id;
```

```
ERROR: Column must be in a GROUP BY clause.
```

Field 3: demographic_group

```sql
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
GROUP BY demographic_group;
```

```
Query Ran Successfully!
```

Field 3: demographic_group

```
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
GROUP BY demographic_group;
```

- No `ELSE` statement = easier to validate

New state of query

```sql
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
GROUP BY demographic_group
UNION ALL
SELECT
    ...
FROM winter_games AS wg
JOIN athletes AS a
ON wg.athlete_id = a.id
GROUP BY demographic_group;
```

Let's practice!

REPORTING IN SQL

Filtering and finishing touches

REPORTING IN SQL

SQL

Tyler Pernes

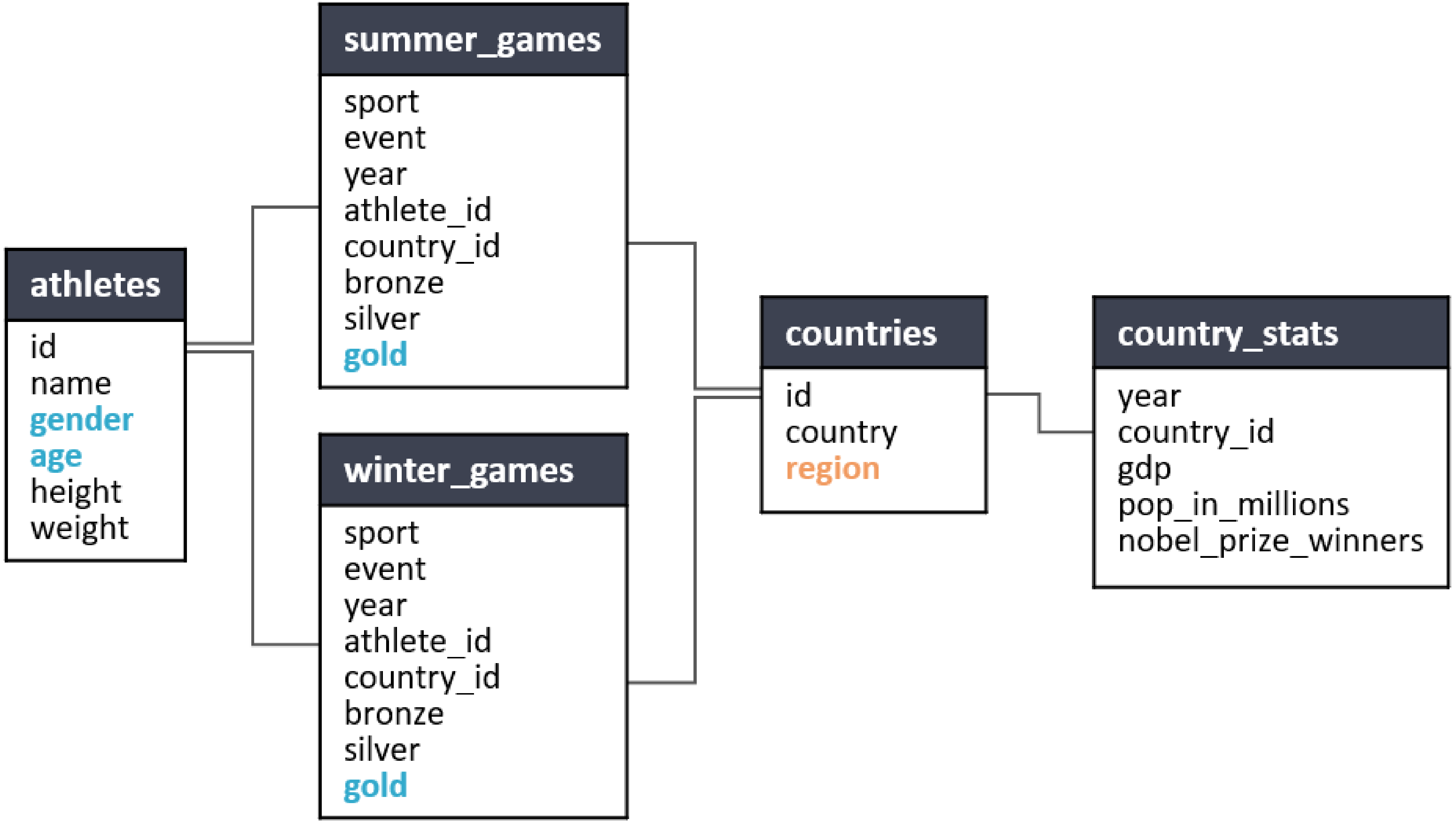
Learning & Development Consultant

datacamp

Goal report

```
Gold Medals by Demographic Group
(Western European Countries Only)
+----------+--------------------+-------+
| season   |  demographic_group | golds |
|----------+--------------------+-------|
| Winter   | Male Age 26+       | 13    |
| Winter   | Female Age 26+     | 8     |
| Summer   | Male Age 13-25     | 7     |
| Summer   | Female Age 13-25   | 6     |
| Winter   | Male Age 13-25     | 4     |
| Summer   | Male Age 26+       | 4     |
| Winter   | Female Age 13-25   | 4     |
| Summer   | Female Age 26+     | 2     |
+----------+--------------------+-------+
```

Filtering



athletes
- id
- name
- gender
- age
- height
- weight

summer_games
- sport
- event
- year
- athlete_id
- country_id
- bronze
- silver
- gold

winter_games
- sport
- event
- year
- athlete_id
- country_id
- bronze
- silver
- gold

countries
- id
- country
- region

country_stats
- year
- country_id
- gdp
- pop_in_millions
- nobel_prize_winners

Filtering with a subquery

Top half of query:

```
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
GROUP BY demographic_group;
```

Filtering with a subquery

Step 1: Setup subquery

```
SELECT id
FROM countries
WHERE region = 'WESTERN EUROPE';
```

```
+-------+
| id    |
|-------|
| 5     |
| 12    |
| 19    |
+-------+
```

Filtering with a subquery

Step 2: Setup `WHERE` statement

```
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
WHERE country_id IN
    (___)
GROUP BY demographic_group;
```

Filtering with a subquery

Step 2: Setup `WHERE` statement

```sql
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
WHERE country_id IN
    (SELECT id
    FROM countries
    WHERE region = 'WESTERN EUROPE')
GROUP BY demographic_group;
```

Filtering with a JOIN

```sql
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
JOIN countries AS c
ON sg.country_id = c.id
WHERE region = 'WESTERN EUROPE'
GROUP BY demographic_group;
```

Remaining questions

- `ORDER BY?`
- `LIMIT?`

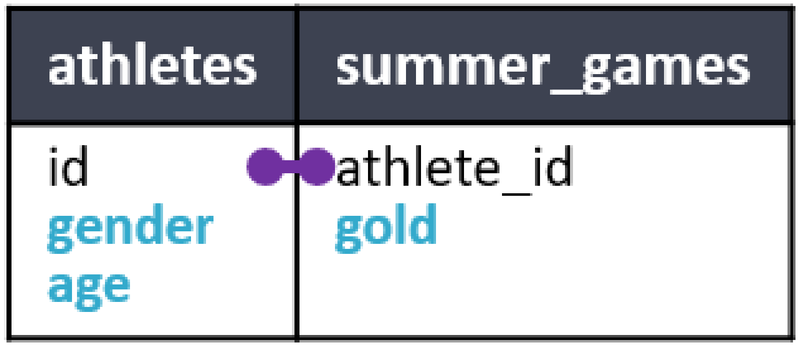
```
Gold Medals by Demographic Group
(Western European Countries Only)
+----------+--------------------+-------+
| season   |  demographic_group | golds |
|----------+--------------------+-------|
| Winter   | Male Age 26+       | 13    |
| Winter   | Female Age 26+     | 8     |
| Summer   | Male Age 13-25     | 7     |
| Summer   | Female Age 13-25   | 6     |
| Winter   | Male Age 13-25     | 4     |
| Summer   | Male Age 26+       | 4     |
| Winter   | Female Age 13-25   | 4     |
| Summer   | Female Age 26+     | 2     |
+----------+--------------------+-------+
```

Final code

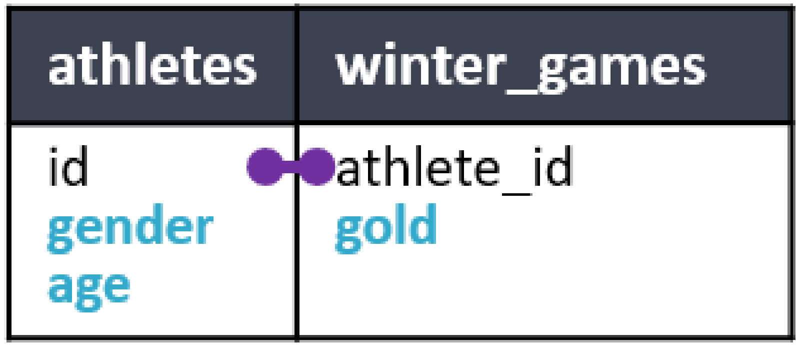
```
SELECT
    'Summer' AS season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) AS golds
FROM summer_games AS sg
JOIN athletes AS a
ON sg.athlete_id = a.id
WHERE country_id IN
    (SELECT id
    FROM countries
    WHERE region = 'WESTERN EUROPE')
GROUP BY demographic_group
UNION ALL
  ...
ORDER BY golds DESC;
```

Order of operations

- Two JOINs



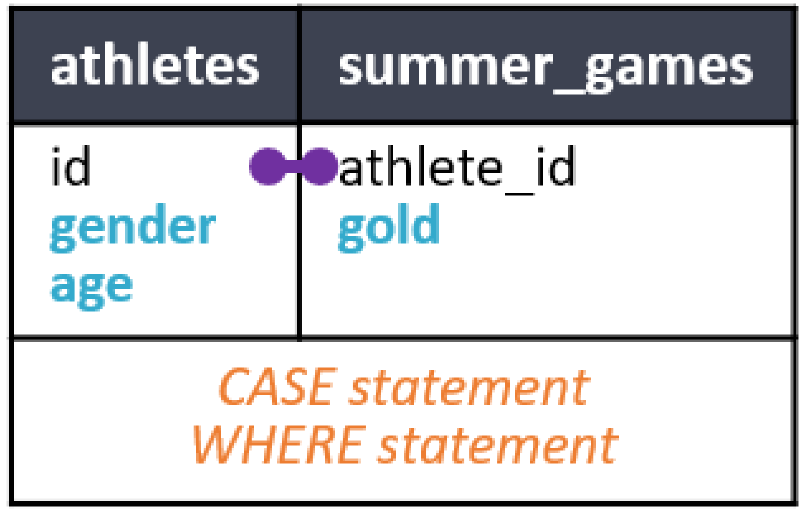
athletes	summer_games
id	athlete_id
gender	gold
age	



athletes	winter_games
id	athlete_id
gender	gold
age	

Order of operations

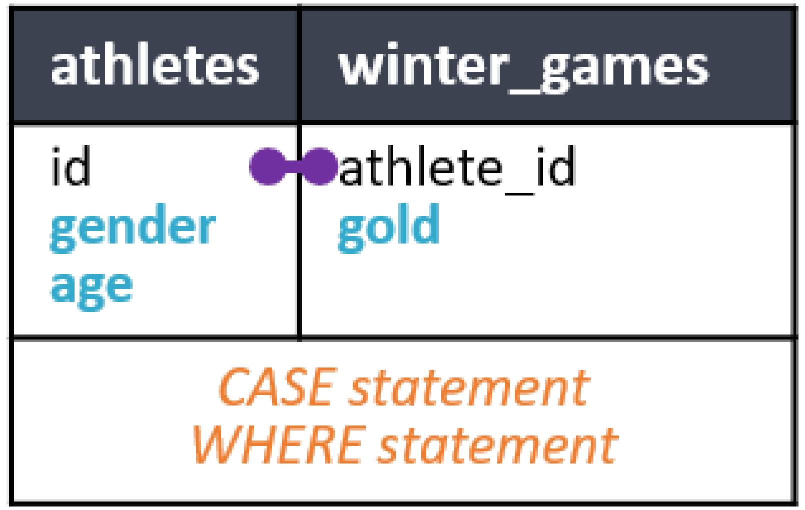
- Two JOINs
- **Add LOGIC**



athletes | summer_games

id, gender, age | athlete_id, gold

CASE statement
WHERE statement



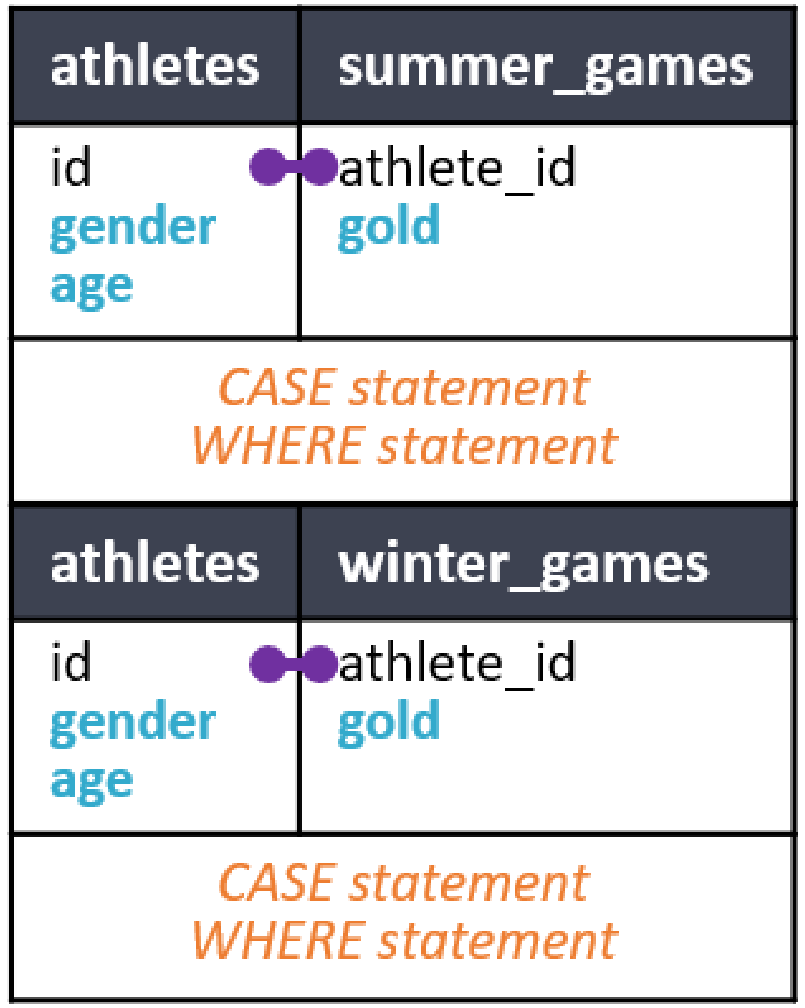
athletes | winter_games

id, gender, age | athlete_id, gold

CASE statement
WHERE statement

Order of operations

- Two JOINs
- Add LOGIC
- **UNION**



athletes	summer_games
id gender age	athlete_id gold

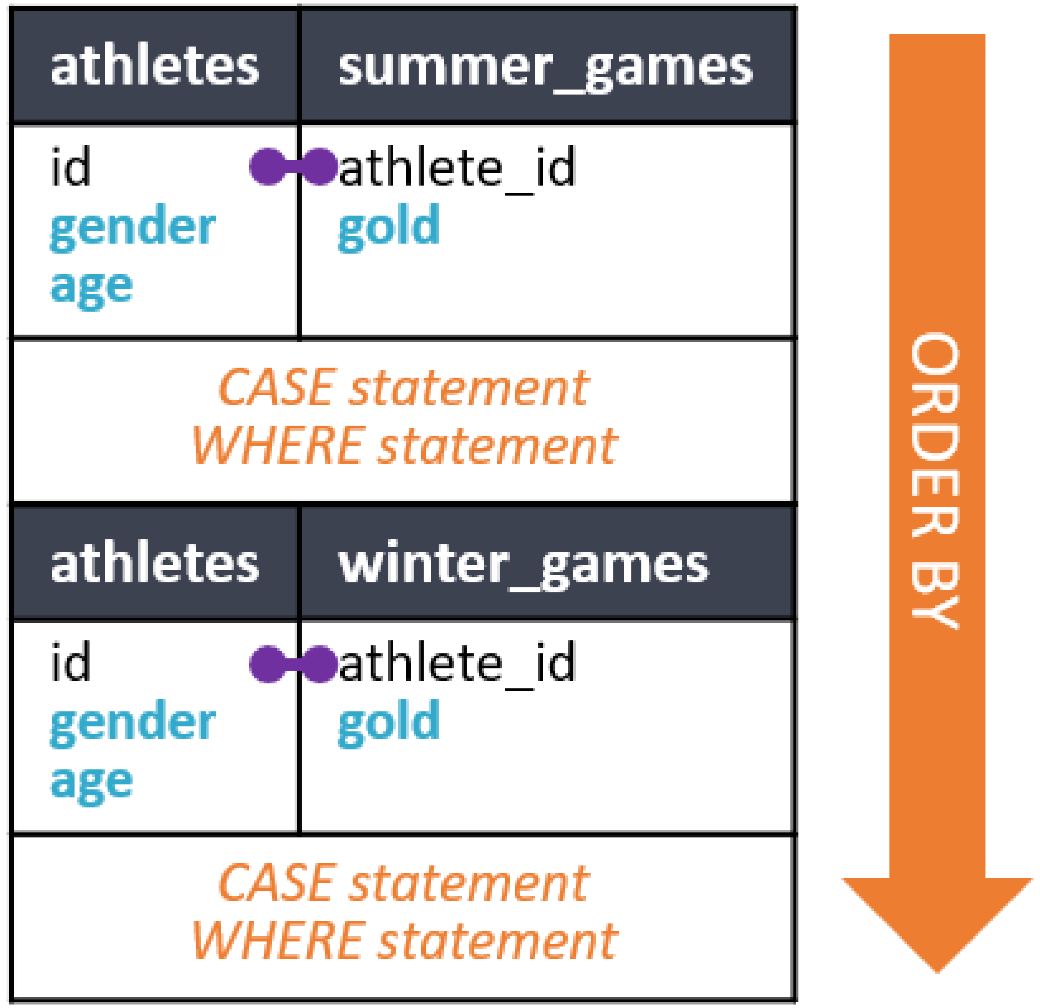
CASE statement
WHERE statement

athletes	winter_games
id gender age	athlete_id gold

CASE statement
WHERE statement

Order of operations

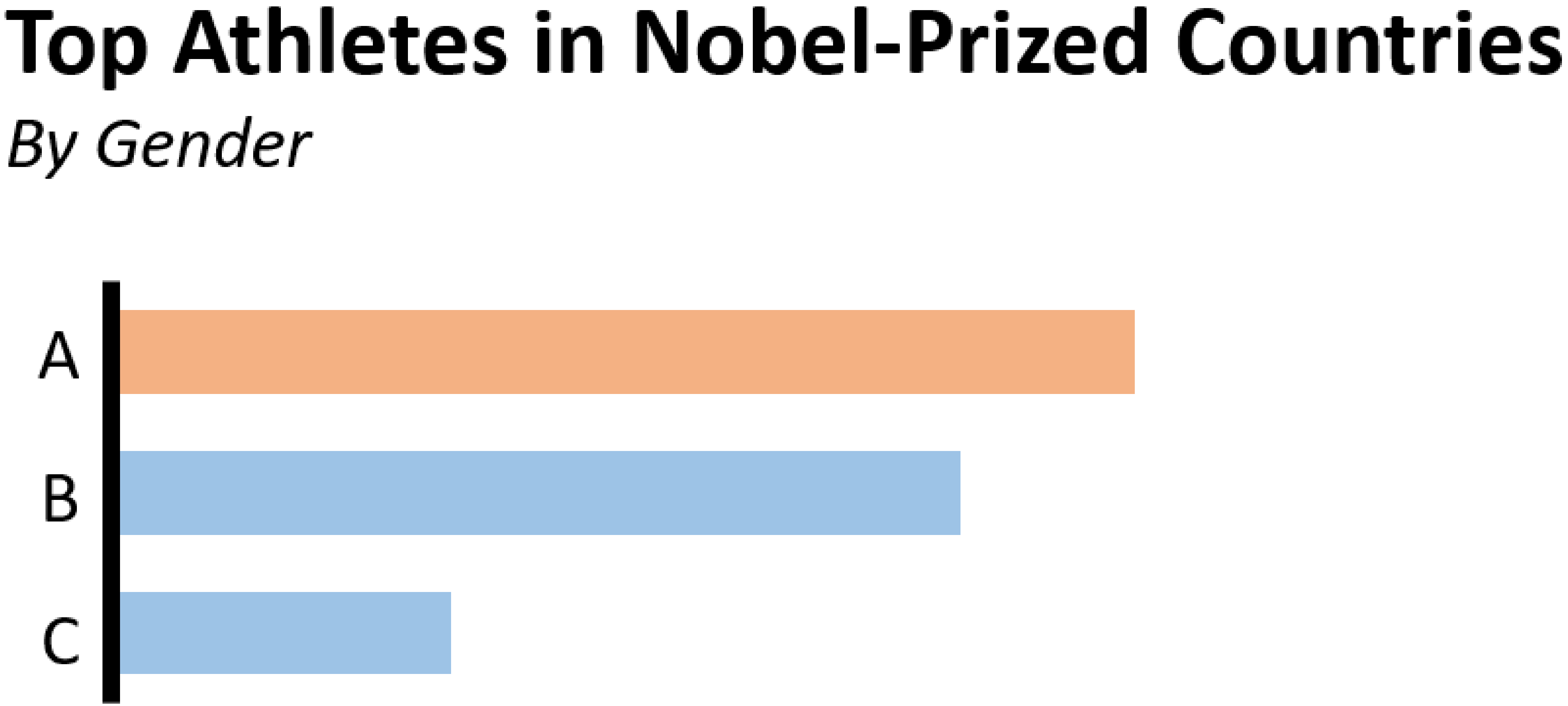
- Two JOINs
- Add LOGIC
- UNION
- **ORDER BY**



Option B

```sql
SELECT
    season,
    CASE WHEN age >= 13 AND age <= 25 AND gender = 'M' THEN 'Male Age 13-25'
    WHEN age > 25 AND gender = 'M' THEN 'Male Age 26+'
    WHEN age >= 13 AND age <= 25 AND gender = 'F' THEN 'Female Age 13-25'
    WHEN age > 25 AND gender = 'F' THEN 'Female Age 26+'
    END AS demographic_group,
    SUM(gold) as golds
FROM
    (SELECT 'Summer' AS season, country_id, athlete_id, gold
    FROM summer_games AS sg
    UNION ALL
    SELECT 'Winter' AS season, country_id, athlete_id, gold
    FROM winter_games AS wg) AS g
JOIN athletes AS a
ON g.athlete_id = a.id
WHERE country_id IN
    (SELECT id
    FROM countries
    WHERE region = 'WESTERN EUROPE')
GROUP BY season, demographic_group
ORDER BY golds DESC;
```

Capstone exercise



Let's practice!

REPORTING IN SQL