Problem solving on Joins

Relevel

by Unacademy

Additional Questions - Set 2 (Joins)

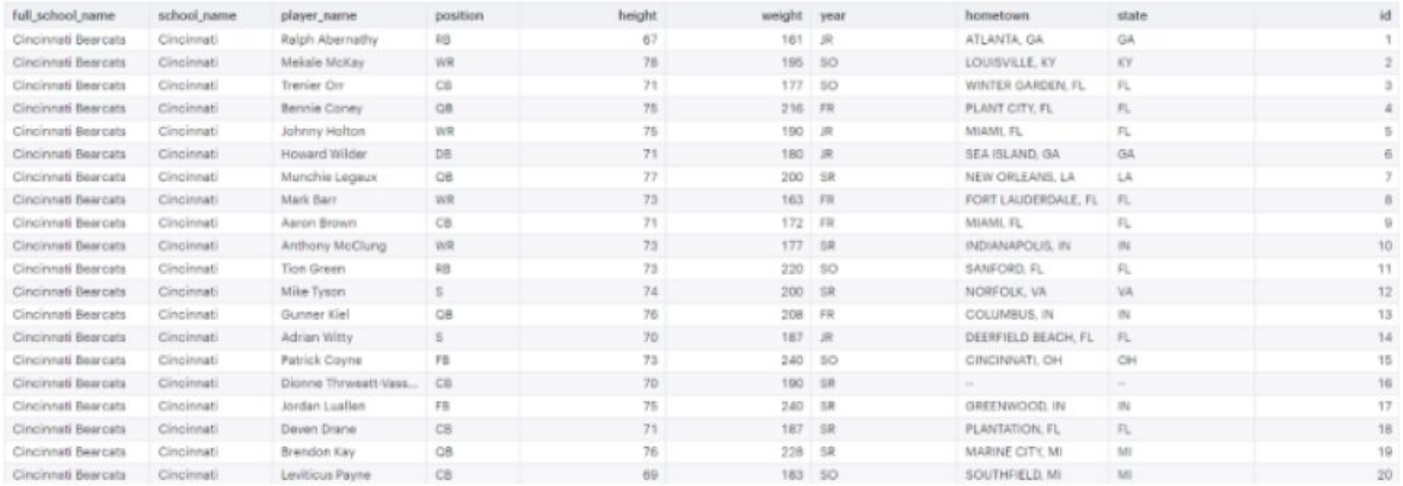
Instructions:

- We will use mode.com for this set of questions.

Caselet - 1

Caselet -1

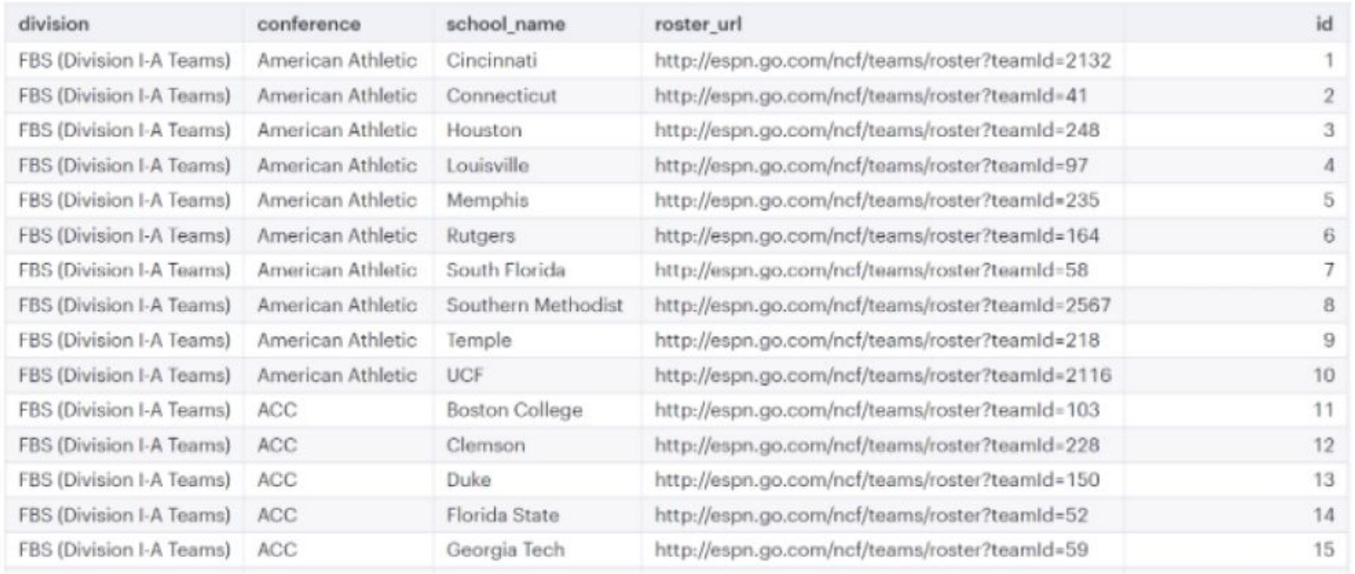
benn.college_football_players players



full_school_name	school_name	player_name	position	height	weight	year	hometown	state	id
Cincinnati Bearcats	Cincinnati	Ralph Abernathy	RB	67	161	JR	ATLANTA, GA	GA	1
Cincinnati Bearcats	Cincinnati	Mekale McKay	WR	78	195	SO	LOUISVILLE, KY	KY	2
Cincinnati Bearcats	Cincinnati	Trenier Orr	CB	71	177	SO	WINTER GARDEN, FL	FL	3
Cincinnati Bearcats	Cincinnati	Bennie Coney	QB	75	216	FR	PLANT CITY, FL	FL	4
Cincinnati Bearcats	Cincinnati	Johnny Holton	WR	75	190	JR	MIAMI, FL	FL	5
Cincinnati Bearcats	Cincinnati	Howard Wilder	DB	71	180	JR	SEA ISLAND, GA	GA	6
Cincinnati Bearcats	Cincinnati	Munchie Legaux	QB	77	200	SR	NEW ORLEANS, LA	LA	7
Cincinnati Bearcats	Cincinnati	Mark Barr	WR	73	163	FR	FORT LAUDERDALE, FL	FL	8
Cincinnati Bearcats	Cincinnati	Aaron Brown	CB	71	172	FR	MIAMI, FL	FL	9
Cincinnati Bearcats	Cincinnati	Anthony McClung	WR	73	177	SR	INDIANAPOLIS, IN	IN	10
Cincinnati Bearcats	Cincinnati	Tion Green	RB	73	220	SO	SANFORD, FL	FL	11
Cincinnati Bearcats	Cincinnati	Mike Tyson	S	74	200	SR	NORFOLK, VA	VA	12
Cincinnati Bearcats	Cincinnati	Gunner Kiel	QB	76	208	FR	COLUMBUS, IN	IN	13
Cincinnati Bearcats	Cincinnati	Adrian Witty	S	70	187	JR	DEERFIELD BEACH, FL	FL	14
Cincinnati Bearcats	Cincinnati	Patrick Coyne	FB	73	240	SO	CINCINNATI, OH	OH	15
Cincinnati Bearcats	Cincinnati	Dionne Thrweatt-Vass...	CB	70	190	SR	--	--	16
Cincinnati Bearcats	Cincinnati	Jordan Luallen	FB	75	240	SR	GREENWOOD, IN	IN	17
Cincinnati Bearcats	Cincinnati	Deven Drane	CB	71	187	SR	PLANTATION, FL	FL	18
Cincinnati Bearcats	Cincinnati	Brendon Kay	QB	76	228	SR	MARINE CITY, MI	MI	19
Cincinnati Bearcats	Cincinnati	Leviticus Payne	CB	69	183	SO	SOUTHFIELD, MI	MI	20

Caselet - 1

benn.college_football_teams



division	conference	school_name	roster_url	id
FBS (Division I-A Teams)	American Athletic	Cincinnati	http://espn.go.com/ncf/teams/roster?teamId=2132	1
FBS (Division I-A Teams)	American Athletic	Connecticut	http://espn.go.com/ncf/teams/roster?teamId=41	2
FBS (Division I-A Teams)	American Athletic	Houston	http://espn.go.com/ncf/teams/roster?teamId=248	3
FBS (Division I-A Teams)	American Athletic	Louisville	http://espn.go.com/ncf/teams/roster?teamId=97	4
FBS (Division I-A Teams)	American Athletic	Memphis	http://espn.go.com/ncf/teams/roster?teamId=235	5
FBS (Division I-A Teams)	American Athletic	Rutgers	http://espn.go.com/ncf/teams/roster?teamId=164	6
FBS (Division I-A Teams)	American Athletic	South Florida	http://espn.go.com/ncf/teams/roster?teamId=58	7
FBS (Division I-A Teams)	American Athletic	Southern Methodist	http://espn.go.com/ncf/teams/roster?teamId=2567	8
FBS (Division I-A Teams)	American Athletic	Temple	http://espn.go.com/ncf/teams/roster?teamId=218	9
FBS (Division I-A Teams)	American Athletic	UCF	http://espn.go.com/ncf/teams/roster?teamId=2116	10
FBS (Division I-A Teams)	ACC	Boston College	http://espn.go.com/ncf/teams/roster?teamId=103	11
FBS (Division I-A Teams)	ACC	Clemson	http://espn.go.com/ncf/teams/roster?teamId=228	12
FBS (Division I-A Teams)	ACC	Duke	http://espn.go.com/ncf/teams/roster?teamId=150	13
FBS (Division I-A Teams)	ACC	Florida State	http://espn.go.com/ncf/teams/roster?teamId=52	14
FBS (Division I-A Teams)	ACC	Georgia Tech	http://espn.go.com/ncf/teams/roster?teamId=59	15

Caselet - 1

Question-1:

Write a query to return player_name, school_name, position, conference from the above dataset.

Caselet - 1

Answer-1:

```
SELECT
  players.player_name,
  players.school_name,
  players.position,
  teams.conference
FROM
  benn.college_football_players players
JOIN
  benn.college_football_teams teams
ON players.school_name = teams.school_name
```

Caselet - 1

Question-2:

Write a query to find the total number of players playing in each conference.Order the output in the descending order of number of players.

Caselet - 1

Answer-2:

```
SELECT
   teams.conference,
   COUNT(players.player_name) AS num_players
FROM
  benn.college_football_players players
JOIN
  benn.college_football_teams teams
ON players.school_name = teams.school_name
GROUP BY
  teams.conference
ORDER BY
  num_players DESC
```

Caselet - 1

Question-3:

Write a query to find the average height of players per division

Caselet - 1

Answer-3:

```
SELECT
    teams.division,
    AVG(players.height) AS avg_height
FROM
  benn.college_football_players players
JOIN
  benn.college_football_teams teams
ON players.school_name = teams.school_name
GROUP BY
  Teams.division
```

Caselet - 1

Question-4:

Write a query to return to the conference where average weight is more than 210. Order the output in the descending order of average weight.

Caselet - 1

Answer-4

```
SELECT
    teams.conference,
    AVG(players.weight) AS avg_weight
FROM
  benn.college_football_players players
JOIN
  benn.college_football_teams teams
ON players.school_name = teams.school_name
GROUP BY
  teams.conference
HAVING
  AVG(players.weight) > 210
ORDER BY
  avg_weight DESC
```

Caselet - 1

Question-5:

Write a query to return to the top 3 conference with the highest BMI (weight/height) ratio

Caselet - 1

Answer-5:

```
SELECT
teams.conference,
703*SUM(players.weight)/SUM(POWER(players.height,2)) AS bmi
FROM
benn.college_football_players players
JOIN
benn.college_football_teams teams
ON players.school_name = teams.school_name
GROUP BY
teams.conference
ORDER BY
bmi DESC
LIMIT 3
```

Caselet - 2

Caselet - 2

- Tutorial.excel_sql_inventory_data
- Tutorial.excel_sql_transaction_data

Caselet - 2

Question-1:

Write a query to join the above tables.

Caselet - 2

Answer-1:

```
SELECT
  a.*,
  b.time
FROM
  tutorial.excel_sql_inventory_data a
LEFT JOIN
  tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
```

Caselet - 2

Question-2:

Find the product which does not sell a single un**it.**

Caselet - 2

Answer-2:

```
SELECT
  a.*,
  b.time
FROM
  tutorial.excel_sql_inventory_data a
LEFT JOIN
  tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
WHERE
  b.time IS NULL
```

Caselet - 2

Question-3:

Write a query to find how many units are sold per product. Sort the data in terms of unit sold(descending order)

Caselet - 2

Answer-3:

```
SELECT
  a.product_id,
  a.product_name,
  COUNT(b.time) AS units_sold
FROM
  tutorial.excel_sql_inventory_data a
LEFT JOIN
  tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
GROUP BY
  a.product_id,
  a.product_name
ORDER BY
  units_sold DESC
```

Caselet - 2

Question-4:

Write a query to return product_type and units_sold where product_type is sold more than 50 times.

Caselet - 2

Answer-4:

```
SELECT
  a.product_type,
  COUNT(b.time) AS units_sold
FROM
  tutorial.excel_sql_inventory_data a
LEFT JOIN
  tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
GROUP BY
  a.product_type
HAVING
  COUNT(b.time) > 50
```

Caselet - 2

Question-5:

Write a query to return the total revenue generated.

Caselet - 2

Answer-5:

```
SELECT
  SUM(price_unit) AS total_revenue
FROM
  tutorial.excel_sql_inventory_data a
LEFT JOIN
  tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
WHERE
  b.time IS NOT NULL
```

Caselet - 2

Question-6:

Write a query to return the most selling product under product_type = ‘dry goods’

Caselet - 2

Answer-6:

```
SELECT
product_name,
COUNT(b.time) AS unit_sold
FROM
tutorial.excel_sql_inventory_data a
LEFT JOIN
tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
WHERE product_type = 'dry_goods'
GROUP BY
product_name
ORDER BY
unit_sold DESC
LIMIT 1
```

Caselet - 2

Question-7:

Write a query to find the difference between inventory and total sales per product_type?

Caselet - 2

Answer-7:

```
SELECT
  product_type,
  SUM(current_inventory) - COUNT(b.time) AS delta
FROM
  tutorial.excel_sql_inventory_data a
LEFT JOIN
  tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
GROUP BY
 product_type
ORDER BY
  delta DESC
```

Caselet - 2

Question-8:

Find the product-wise sales for product_type =’dairy’

Caselet - 2

Answer-8:

```
SELECT
 a.product_name,
 SUM(a.price_unit)*COUNT(b.time) AS sales
FROM
  tutorial.excel_sql_inventory_data a
LEFT JOIN
  tutorial.excel_sql_transaction_data b
ON a.product_id = b.product_id
WHERE
  product_type = 'dairy'
GROUP BY
  product_name
ORDER BY
  sales DESC
```

Caselet - 3

Caselet - 3

- Tutorial.yammer_users
- Tutorial.yammer_experiments
- Tutorial.yammer_events
- Tutorial.yammer_emails

Caselet - 3

Question-1:

Find the number of users per language type/

Caselet - 3

Anwer-1:

```
SELECT
  language,
  COUNT(user_id) AS num_user
FROM
  tutorial.yammer_users
GROUP BY
  language
```

Caselet - 3

Question-2:

Write a query to find how many users are part of experiments.

Caselet - 3

Answer-2:

```
SELECT
  COUNT(DISTINCT a.user_id) AS total_users,
  COUNT(DISTINCT b.user_id) AS users_experiment
FROM
  tutorial.yammer_users a
LEFT JOIN
  tutorial.yammer_events b
ON a.user_id = b.user_id
```

Caselet - 3

Question-3:

Find the number of users in experiment per language category.

Caselet - 3

Answer-3:

```
SELECT
  a.language,
  COUNT(DISTINCT b.user_id) AS users_experiment
FROM
  tutorial.yammer_users a
LEFT JOIN
  tutorial.yammer_events b
ON a.user_id = b.user_id
GROUP BY
  a.language
```

Caselet - 3

Question-4:

Write a query to find how many users have received at least one email

Caselet - 3

Answer-4:

```
SELECT

  COUNT(DISTINCT a.user_id) AS total_users,

  COUNT(DISTINCT b.user_id) AS users_with_email

FROM

  tutorial.yammer_users a

LEFT JOIN

  tutorial.yammer_emails b

ON a.user_id = b.user_id
```

Caselet - 3

Question-5:

Write a query to find how many users per company id have received at least one email?

Caselet - 3

Answer-5:

```
SELECT
  a.company_id,
  COUNT(DISTINCT b.user_id) AS users_with_email
FROM
  tutorial.yammer_users a
LEFT JOIN
  tutorial.yammer_emails b
ON a.user_id = b.user_id
GROUP BY
  a.company_id
```

Caselet - 3

Question-6:

Write a query to find how many users have received at least one event

Caselet - 3

Answer-6:

```
SELECT
  COUNT(DISTINCT a.user_id) AS total_users,
  COUNT(DISTINCT b.user_id) AS users_with_events
FROM
  tutorial.yammer_users a
LEFT JOIN
  tutorial.yammer_events b
ON a.user_id = b.user_id
```

Caselet - 3

Question-7:

Write a query to find how many distinct users per state have at least one event?

Caselet - 3

Answer-7:

```
SELECT
  a.state,
  COUNT(DISTINCT b.user_id) AS users_with_events
FROM
  tutorial.yammer_users a
LEFT JOIN
  tutorial.yammer_events b
ON a.user_id = b.user_id
GROUP BY
  a.state
```

Caselet - 4

Caselet - 4

Question-1:

Write a query to join the tables tutorial.us_housing_units and tutorial.us_housing_units_completed. Return all the records

Caselet - 4

Answer-1:

```
SELECT
a.year,
a.month,
a.month_name,
a.south AS south_unit,
a.west AS west_unit,
a.midwest AS midwest_unit,
a.northeast AS northeast_unit,
b.south AS south_completed,
b.west AS west_completed,
b.midwest AS midwest_completed,
b.northeast AS northeast_completed
FROM
tutorial.us_housing_units a
LEFT JOIN
tutorial.us_housing_units_completed b
ON a.year = b.year
AND a.month = b.month
```

Caselet - 4

Question-2:

Write a query to return year, month, month_name and difference between the units and units completed for west from 2000 onwards.

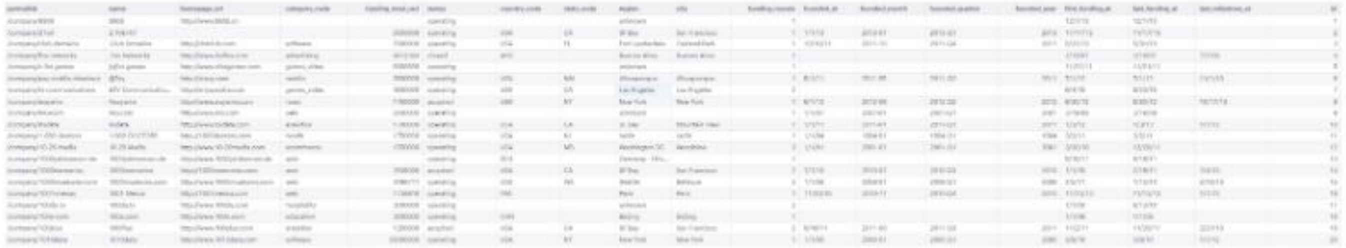
Caselet - 4

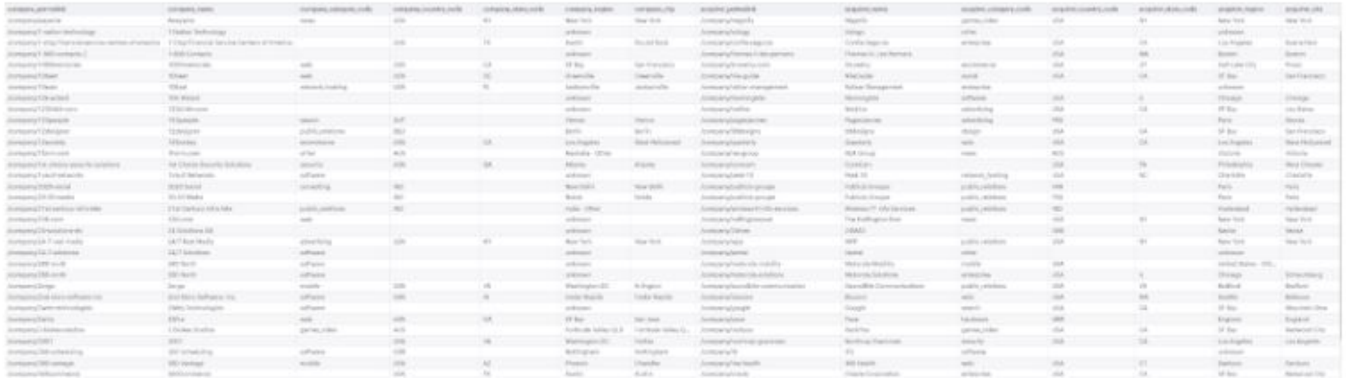
Answer-2:

```
SELECT
a.year,
a.month,
a.month_name,
a.west as west_a,
b.west as west_b,
a.west - b.west AS difference_in_units
FROM
tutorial.us_housing_units a
LEFT JOIN
tutorial.us_housing_units_completed b
ON a.year = b.year
AND a.month = b.month
WHERE
a.year >= 2000
```

Caselet - 5

Caselet - 5


[illegible]


[illegible]

#270DaysofPurpose

Relevel
by Unacademy

Caselet - 5

Question-1:

Write a query that performs a left join between the tutorial.crunchbase_companies table and tutorial.crunchbase_acquisitions table. List the individual rows.

Caselet - 5

Answer-1:

```
SELECT *
  FROM tutorial.crunchbase_companies companies
  LEFT JOIN tutorial.crunchbase_acquisitions acquisitions
    ON companies.permalink = acquisitions.company_permalink
```

Caselet - 5

Question-2:

Count the number of unique companies (don't double-count companies) and unique acquired companies.

Caselet - 5

Answer-2

```
SELECT

    COUNT(DISTINCT companies.permalink) AS unique_companies,

    COUNT(DISTINCT acquisitions.company_permalink) AS unique_companies_acquired

 FROM tutorial.crunchbase_companies companies

 LEFT JOIN tutorial.crunchbase_acquisitions acquisitions

  ON companies.permalink = acquisitions.company_permalink
```

Caselet - 5

Question-3:

Write a query to give a count of number of companies which never acquired any company

Caselet - 5

Answer-3:

```
SELECT  COUNT( DISTINCT companies.permalink) AS num_no_acquisition

FROM

  tutorial.crunchbase_companies companies

LEFT JOIN

  tutorial.crunchbase_acquisitions acquisitions

ON companies.permalink = acquisitions.company_permalink

WHERE

  acquisitions.company_permalink IS NULL
```

Caselet - 5

Question-4:

Count the number of unique companies (don't double-count companies) and unique acquired companies by state. Do not include results for which there is no state data, and order by the number of acquired companies from highest to lowest.

Caselet - 5

Answer-4:

```
SELECT companies.state_code,
       COUNT(DISTINCT companies.permalink) AS unique_companies,
       COUNT(DISTINCT acquisitions.company_permalink) AS unique_companies_acquired
  FROM tutorial.crunchbase_companies companies
  LEFT JOIN tutorial.crunchbase_acquisitions acquisitions
    ON companies.permalink = acquisitions.company_permalink
 WHERE companies.state_code IS NOT NULL
 GROUP BY 1
 ORDER BY 3 DESC
```

Caselet - 5

Question-5:

Write a query that joins tutorial.crunchbase_companies and tutorial.crunchbase_investments_part1 using a FULL JOIN. Count up the number of rows that are present in one table and present in both the table.

Caselet - 5

Answer-5:

```
SELECT
COUNT(CASE WHEN companies.permalink IS NOT NULL AND investments.company_permalink IS NULL
                  THEN companies.permalink ELSE NULL END) AS companies_only,
             COUNT(CASE WHEN companies.permalink IS NOT NULL AND investments.company_permalink IS NOT
NULL
                  THEN companies.permalink ELSE NULL END) AS both_tables,
         COUNT(CASE WHEN companies.permalink IS NULL AND investments.company_permalink IS NOT NULL
                  THEN investments.company_permalink ELSE NULL END) AS investments_only
     FROM
tutorial.crunchbase_companies companies
     FULL JOIN
tutorial.crunchbase_investments_part1 investments
       ON companies.permalink = investments.company_permalink
```

Caselet - 5

Question-6:

Write a query to find the records where a company received investment 5 year after founding year.

Caselet - 5

Answer-6:

```
SELECT companies.permalink,
       companies.name,
       companies.status,
       COUNT(investments.investor_permalink) AS investors
  FROM
tutorial.crunchbase_companies companies
  LEFT JOIN
tutorial.crunchbase_investments_part1 investments
    ON companies.permalink = investments.company_permalink
   AND investments.funded_year > companies.founded_year + 5
 GROUP BY 1,2, 3
```

Caselet - 7

Caselet - 7

Question-1:

In the tutorial.city_populations dataset, add a column which tells how many cities have less population than the city mentioned in the row

Caselet - 7

Answer-1:

```
SELECT
  a.city,
  a.state,
  a.population_estimate_2012,
  COUNT(b.city) AS num_city_with_higher_population
FROM
  tutorial.city_populations a
JOIN
  tutorial.city_populations b
ON a.population_estimate_2012 >b.population_estimate_2012
GROUP BY
  a.city,
  a.state,
  A.population_estimate_2012
```

Caselet - 7

Question-2:

In the tutorial.city_populations dataset, add a column which tells the rank of city in terms of population. City with highest population should get rank = 1

Caselet - 7

Answer-2:

```
SELECT
  a.city,
  a.state,
  a.population_estimate_2012,
  COUNT(b.city) AS rank
FROM
  tutorial.city_populations a
JOIN
  tutorial.city_populations b
ON a.population_estimate_2012 <=b.population_estimate_2012
GROUP BY
  a.city,
  a.state,
  a.population_estimate_2012
ORDER BY
  rank
```

THANK YOU