DEALING WITH DATA

SPRING 2019: INFO-GB.2346.30

PROFESSOR GUTHRIE COLLIN

TEACHING FELLOW AJINKYA WALIMBE

NYU | STERN

CLASS 7: SQL

APRIL 4, 2019

CONTENTS

1. SUBQUERIES and WITH

SUBQUERIES

- Subqueries are temporary tables created with nested SELECT statements where a table should be, typically used in JOINS
- Subqueries can enable deeper analysis with SQL as JOINS or WHERE conditionals
- Subqueries (and nesting them) is fast, easy to code-up once, but can be slow, hard to maintain, and can come with a cost (in query performance)
- Two ways to improve these: (1) Add Aliases; (2) Use WITH clause

NESTED *subqueries* AND *aliases*

```
SELECT
    ...
FROM Table1
LEFT JOIN (
         SELECT ...
             FROM (
                  SELECT ...
                  FROM Table3
                  RIGHT JOIN ( ... )
                  )
         )
WHERE Column IN (SELECT … FROM …)
```

SUBQUERY in JOIN

SUBQUERY in WHERE conditional

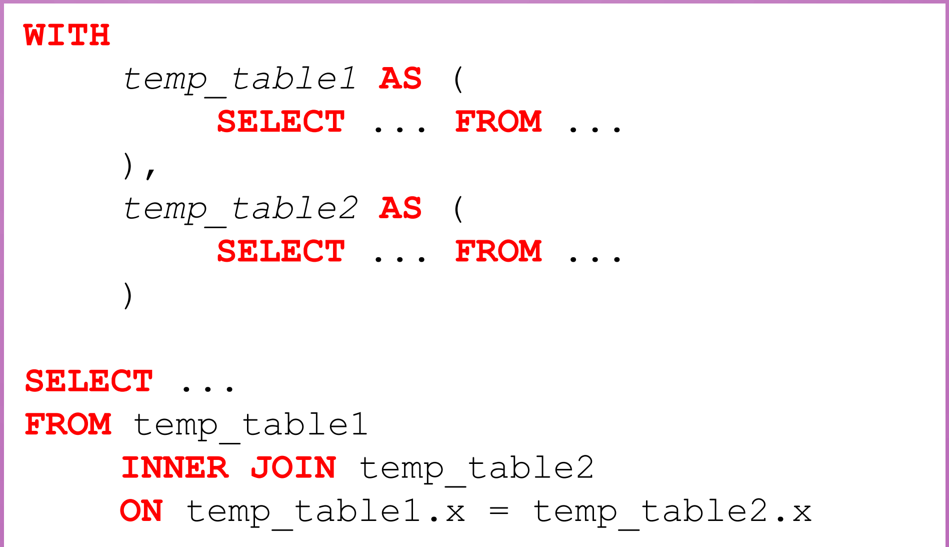
```
SELECT
    ...
FROM Table1
LEFT JOIN (
    SELECT ...
    FROM Table2
) AS t2 ON Table1.x = t2.x
INNER JOIN (
    SELECT ...
    FROM Table3
) AS t3 ON Table1.y = t3.y
```

SUBQUERY alias

WITH CLAUSE

- The WITH clause creates a Common Table Expression and enables sub-query empowered analyses with easier SQL codebase

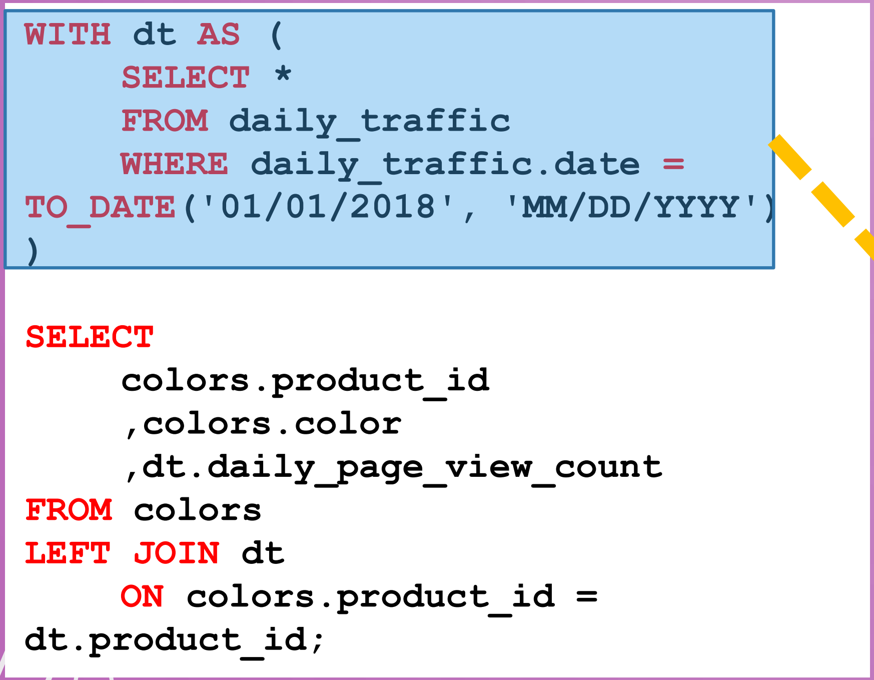
- Syntax:



```
WITH
     temp_table1 AS (
          SELECT ... FROM ...
     ),
     temp_table2 AS (
          SELECT ... FROM ...
     )

SELECT ...
FROM temp_table1
     INNER JOIN temp_table2
     ON temp_table1.x = temp_table2.x
```

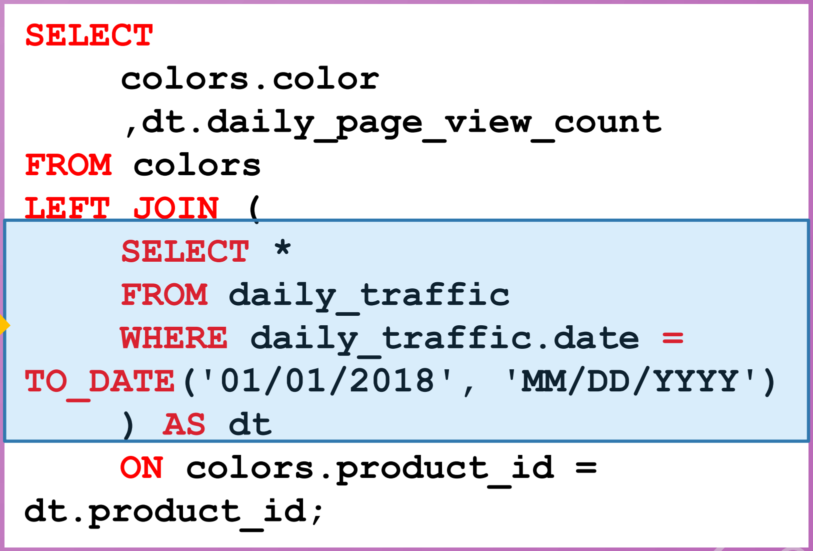
SAME QUERY,
TWO SUBQUERY SOLUTIONS

WITH *common_table_expression*
AS *(query)*



```
WITH dt AS (
     SELECT *
     FROM daily_traffic
     WHERE daily_traffic.date =
TO_DATE('01/01/2018', 'MM/DD/YYYY')
)

SELECT
     colors.product_id
     ,colors.color
     ,dt.daily_page_view_count
FROM colors
LEFT JOIN dt
     ON colors.product_id =
dt.product_id;
```

Nested *(subquery)* **AS** *alias*



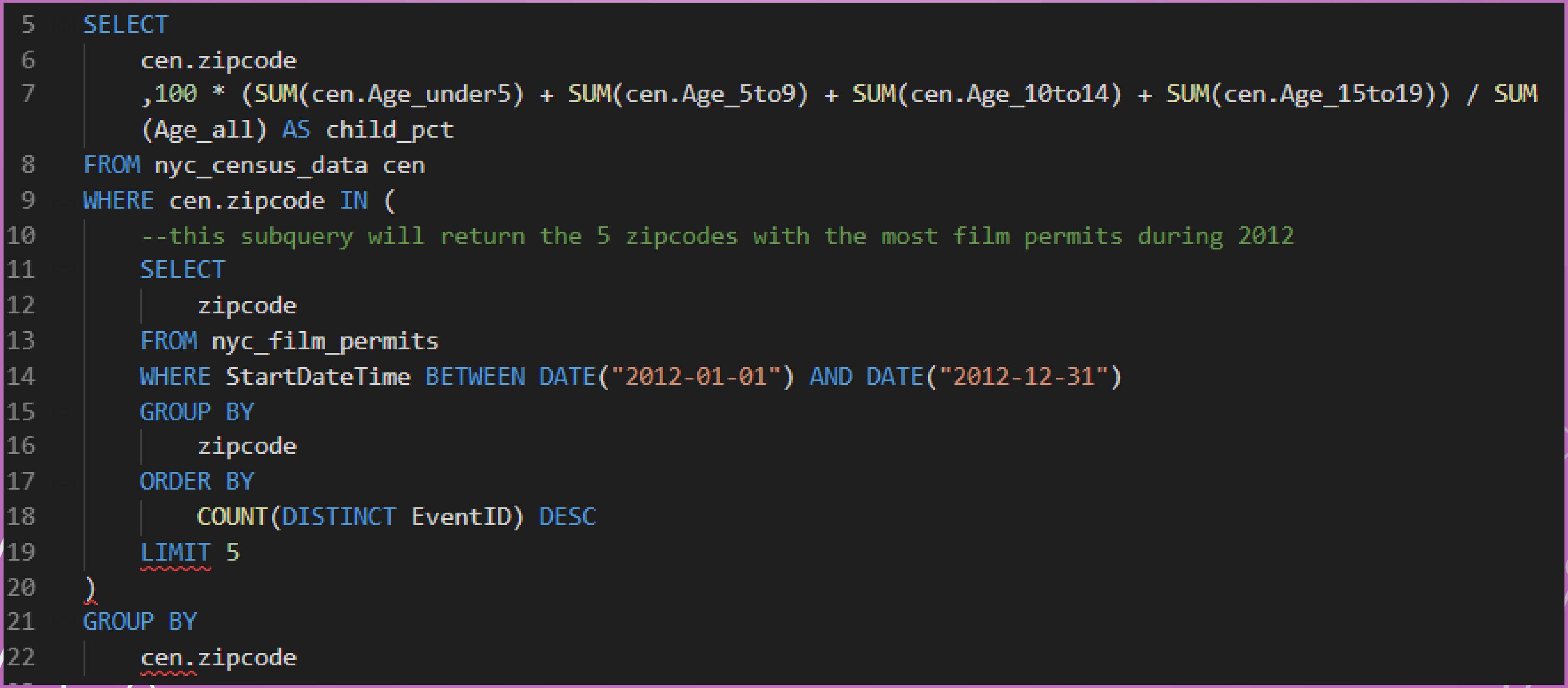
```
SELECT
     colors.color
     ,dt.daily_page_view_count
FROM colors
LEFT JOIN (
     SELECT *
     FROM daily_traffic
     WHERE daily_traffic.date =
TO_DATE('01/01/2018', 'MM/DD/YYYY')
     ) AS dt
     ON colors.product_id =
dt.product_id;
```

SUBQUERY PRACTICE #1 –
SUBQUERY IN WHERE CONDITIONAL

Find the % of people under 19 years old in the 5 zipcodes with the most film permits during 2012

SUBQUERY PRACTICE #1 – SUBQUERY IN WHERE CONDITIONAL

Find the % of people under 19 years old in the 5 zipcodes with the most film permits during 2012

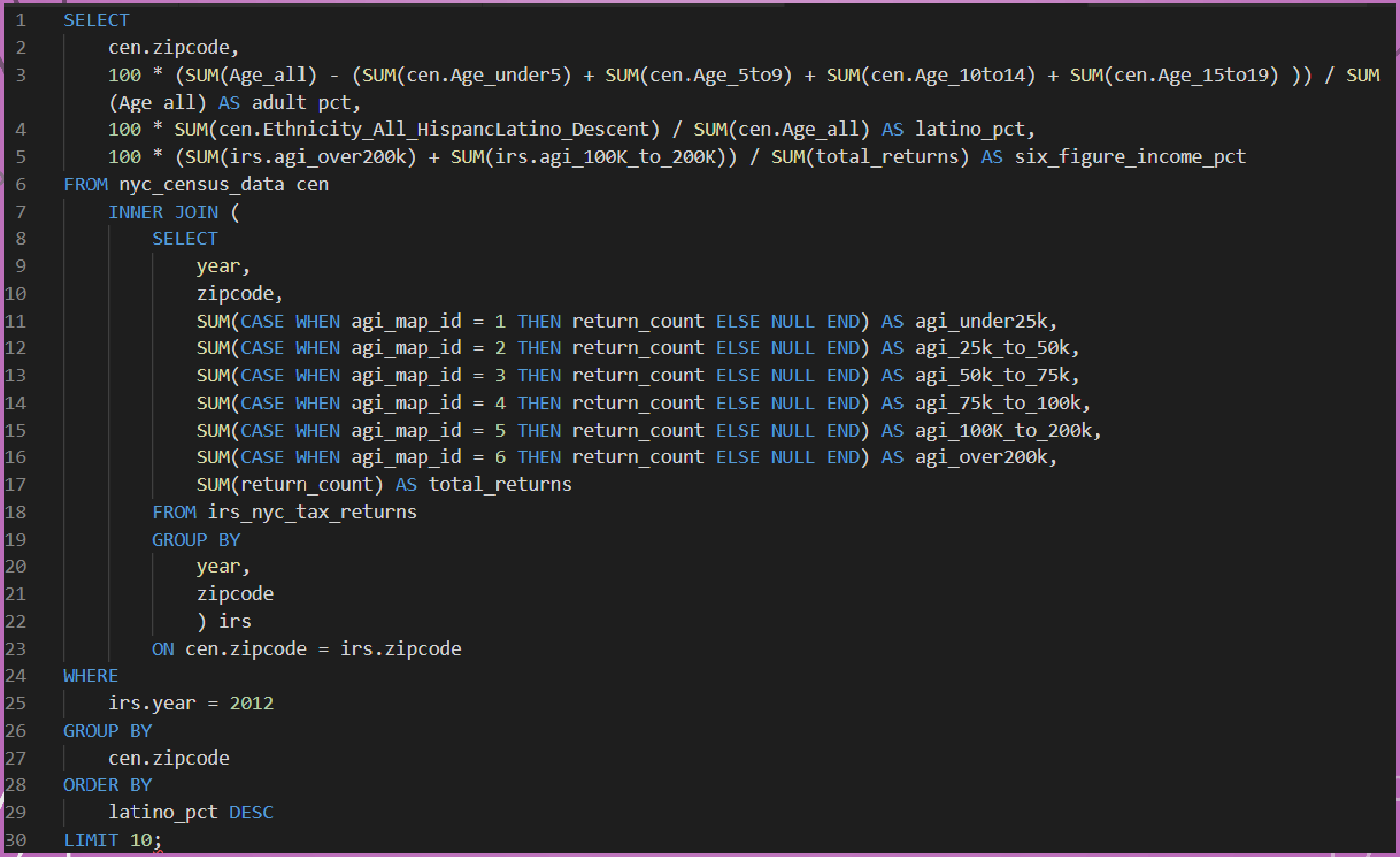


```
SELECT
    cen.zipcode
    ,100 * (SUM(cen.Age_under5) + SUM(cen.Age_5to9) + SUM(cen.Age_10to14) + SUM(cen.Age_15to19)) / SUM(Age_all) AS child_pct
FROM nyc_census_data cen
WHERE cen.zipcode IN (
    --this subquery will return the 5 zipcodes with the most film permits during 2012
    SELECT
        zipcode
    FROM nyc_film_permits
    WHERE StartDateTime BETWEEN DATE("2012-01-01") AND DATE("2012-12-31")
    GROUP BY
        zipcode
    ORDER BY
        COUNT(DISTINCT EventID) DESC
    LIMIT 5
)
GROUP BY
    cen.zipcode
```

SUBQUERY PRACTICE #2 –
2A: NESTED VS 2B: WITH CTE

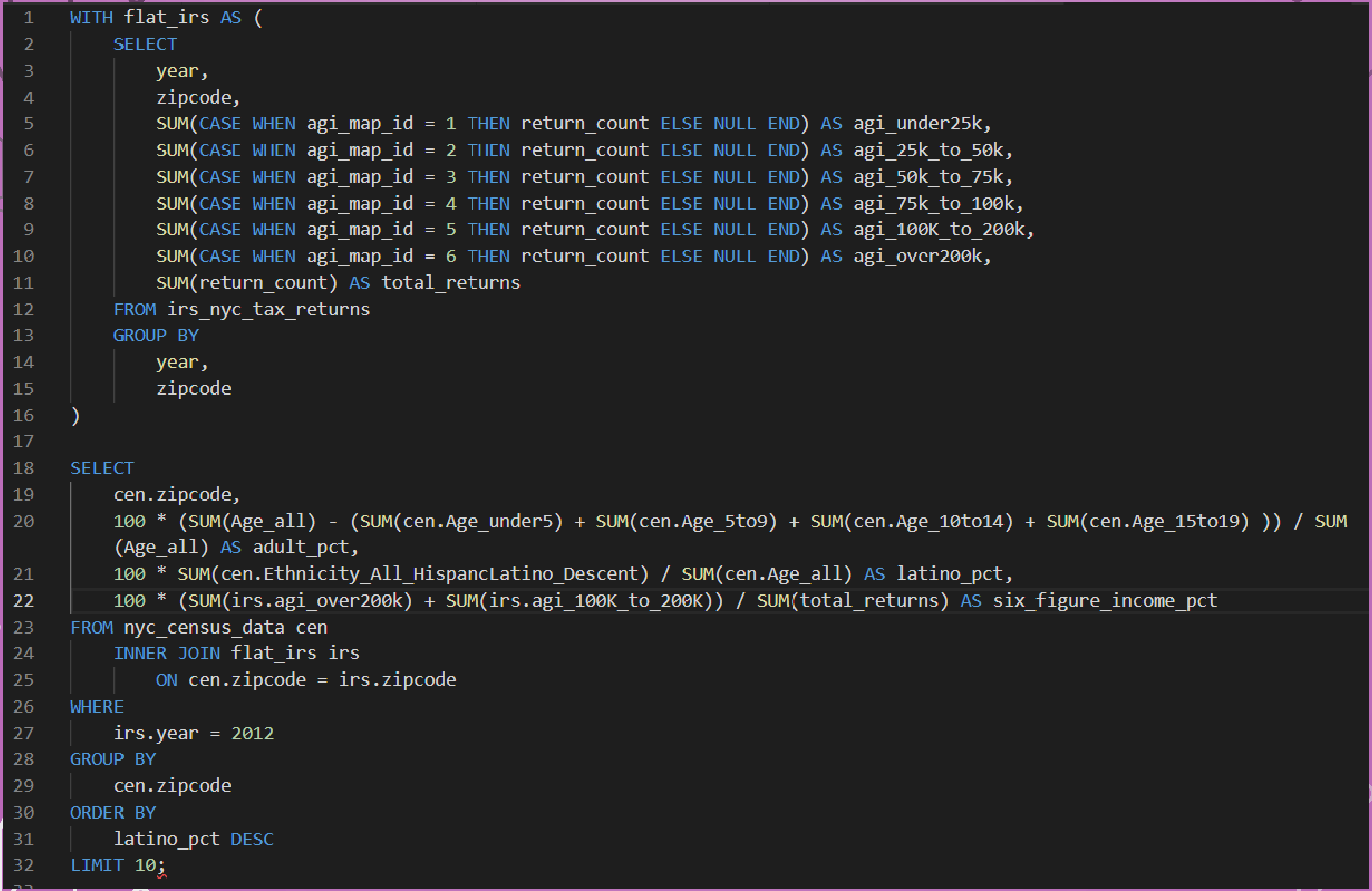
Find the top 10 zipcodes with the largest percentage of persons of Latino/Hispanic descent, and calculate the percentage of adults and percentage of six-figure income earners in these zip codes

SUBQUERY PRACTICE #2A – NESTED



```sql
SELECT
    cen.zipcode,
    100 * (SUM(Age_all) - (SUM(cen.Age_under5) + SUM(cen.Age_5to9) + SUM(cen.Age_10to14) + SUM(cen.Age_15to19) )) / SUM(Age_all) AS adult_pct,
    100 * SUM(cen.Ethnicity_All_HispancLatino_Descent) / SUM(cen.Age_all) AS latino_pct,
    100 * (SUM(irs.agi_over200k) + SUM(irs.agi_100K_to_200K)) / SUM(total_returns) AS six_figure_income_pct
FROM nyc_census_data cen
    INNER JOIN (
        SELECT
            year,
            zipcode,
            SUM(CASE WHEN agi_map_id = 1 THEN return_count ELSE NULL END) AS agi_under25k,
            SUM(CASE WHEN agi_map_id = 2 THEN return_count ELSE NULL END) AS agi_25k_to_50k,
            SUM(CASE WHEN agi_map_id = 3 THEN return_count ELSE NULL END) AS agi_50k_to_75k,
            SUM(CASE WHEN agi_map_id = 4 THEN return_count ELSE NULL END) AS agi_75k_to_100k,
            SUM(CASE WHEN agi_map_id = 5 THEN return_count ELSE NULL END) AS agi_100K_to_200k,
            SUM(CASE WHEN agi_map_id = 6 THEN return_count ELSE NULL END) AS agi_over200k,
            SUM(return_count) AS total_returns
        FROM irs_nyc_tax_returns
        GROUP BY
            year,
            zipcode
            ) irs
        ON cen.zipcode = irs.zipcode
WHERE
    irs.year = 2012
GROUP BY
    cen.zipcode
ORDER BY
    latino_pct DESC
LIMIT 10;
```

SUBQUERY PRACTICE #2B – WITH CTE

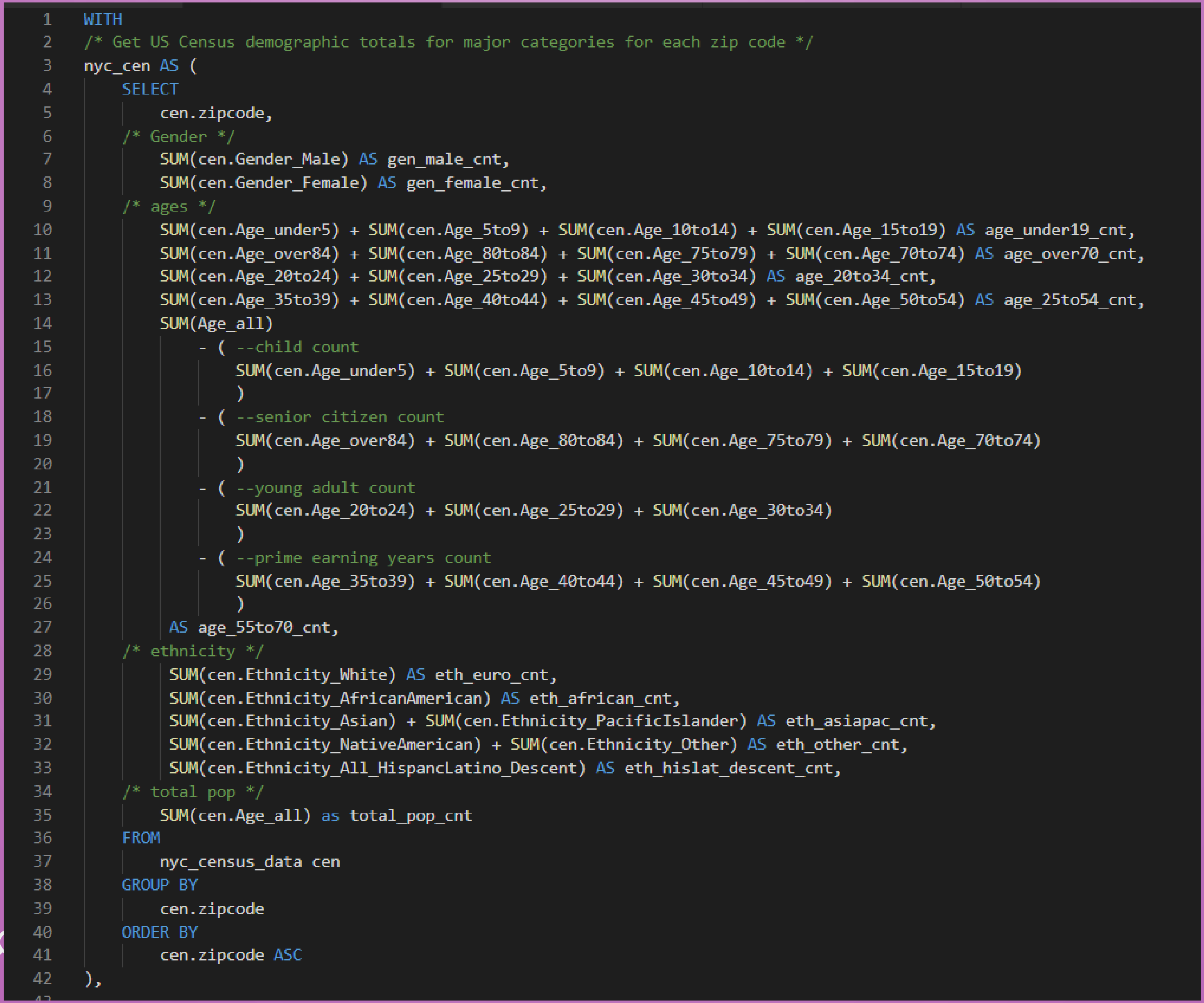


```sql
WITH flat_irs AS (
    SELECT
        year,
        zipcode,
        SUM(CASE WHEN agi_map_id = 1 THEN return_count ELSE NULL END) AS agi_under25k,
        SUM(CASE WHEN agi_map_id = 2 THEN return_count ELSE NULL END) AS agi_25k_to_50k,
        SUM(CASE WHEN agi_map_id = 3 THEN return_count ELSE NULL END) AS agi_50k_to_75k,
        SUM(CASE WHEN agi_map_id = 4 THEN return_count ELSE NULL END) AS agi_75k_to_100k,
        SUM(CASE WHEN agi_map_id = 5 THEN return_count ELSE NULL END) AS agi_100K_to_200k,
        SUM(CASE WHEN agi_map_id = 6 THEN return_count ELSE NULL END) AS agi_over200k,
        SUM(return_count) AS total_returns
    FROM irs_nyc_tax_returns
    GROUP BY
        year,
        zipcode
)

SELECT
    cen.zipcode,
    100 * (SUM(Age_all) - (SUM(cen.Age_under5) + SUM(cen.Age_5to9) + SUM(cen.Age_10to14) + SUM(cen.Age_15to19) )) / SUM(Age_all) AS adult_pct,
    100 * SUM(cen.Ethnicity_All_HispancLatino_Descent) / SUM(cen.Age_all) AS latino_pct,
    100 * (SUM(irs.agi_over200k) + SUM(irs.agi_100K_to_200K)) / SUM(total_returns) AS six_figure_income_pct
FROM nyc_census_data cen
    INNER JOIN flat_irs irs
        ON cen.zipcode = irs.zipcode
WHERE
    irs.year = 2012
GROUP BY
    cen.zipcode
ORDER BY
    latino_pct DESC
LIMIT 10;
```

SUBQUERY PRACTICE #3 – WITH C*TE*

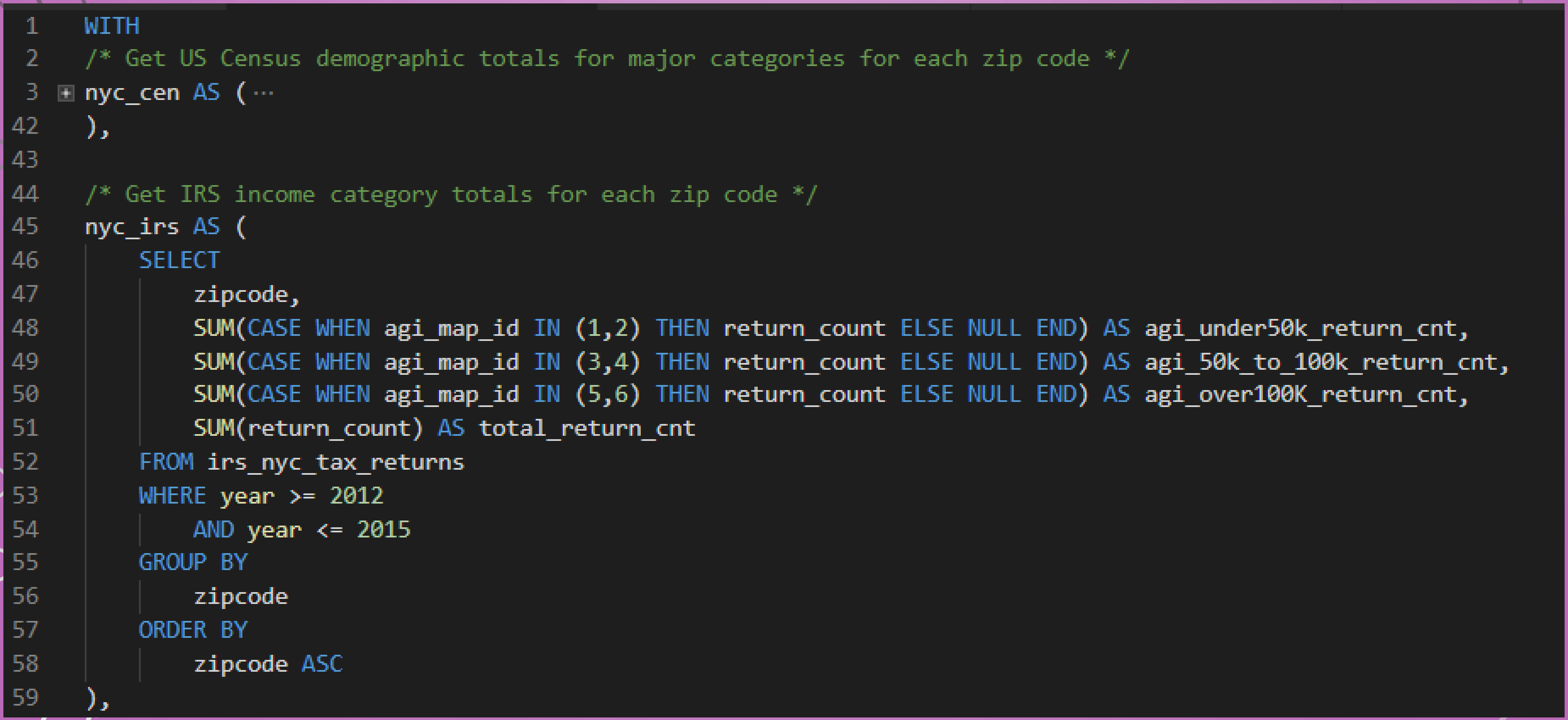
CLASS PROJECT ➔ we want to compare the demographic makeup of NYC versus the demographic makeup of the “NYC on video media (films/TV)”. To do this, we first need to understand the percentage of NYC citizens in each major demographic bucket to compare versus the films.

SUBQUERY PRACTICE #3 – WITH *CTE*, *PART 1 ➔ CTE FOR CENSUS DATA*



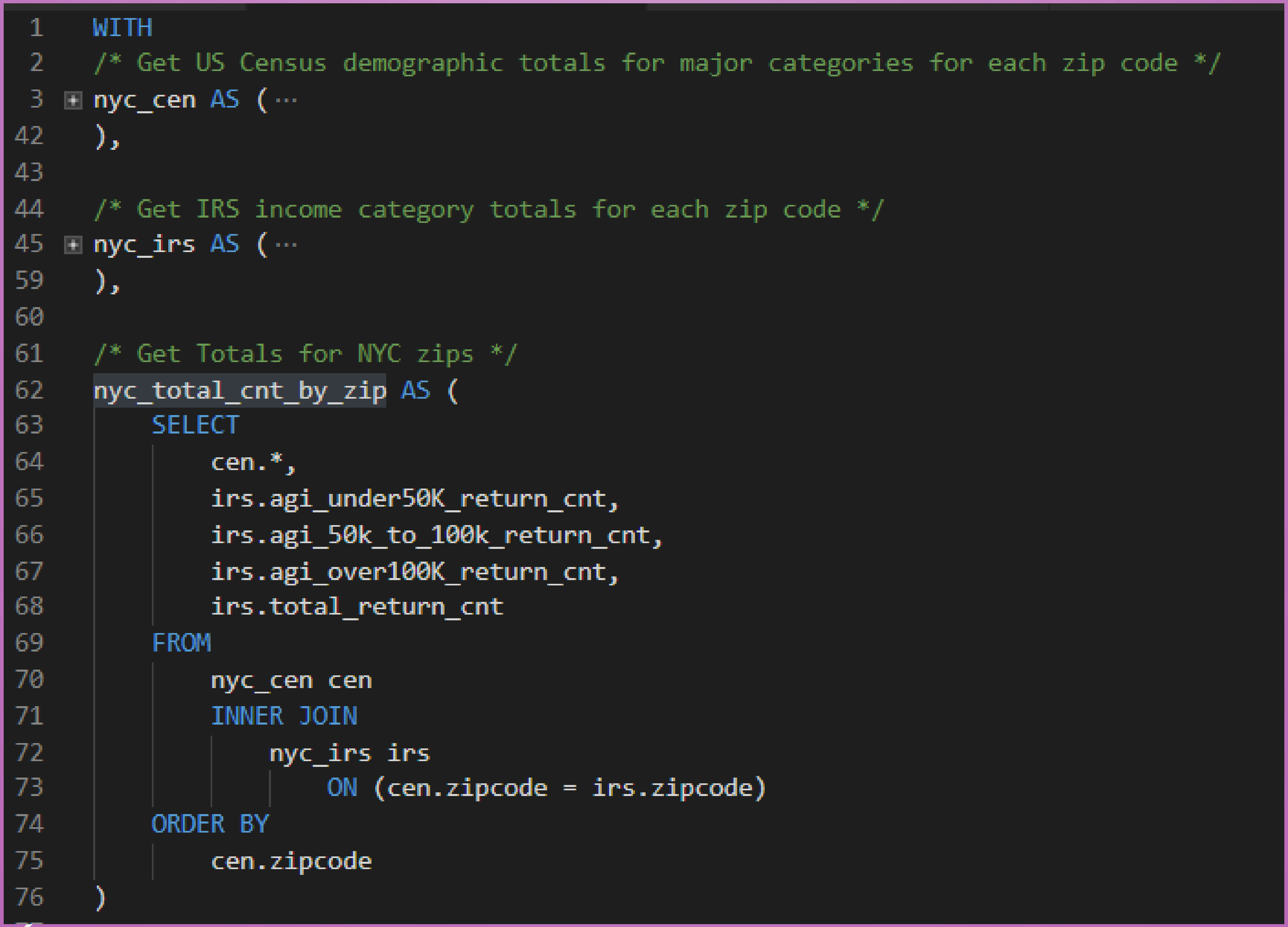
```sql
WITH
/* Get US Census demographic totals for major categories for each zip code */
nyc_cen AS (
    SELECT
        cen.zipcode,
    /* Gender */
        SUM(cen.Gender_Male) AS gen_male_cnt,
        SUM(cen.Gender_Female) AS gen_female_cnt,
    /* ages */
        SUM(cen.Age_under5) + SUM(cen.Age_5to9) + SUM(cen.Age_10to14) + SUM(cen.Age_15to19) AS age_under19_cnt,
        SUM(cen.Age_over84) + SUM(cen.Age_80to84) + SUM(cen.Age_75to79) + SUM(cen.Age_70to74) AS age_over70_cnt,
        SUM(cen.Age_20to24) + SUM(cen.Age_25to29) + SUM(cen.Age_30to34) AS age_20to34_cnt,
        SUM(cen.Age_35to39) + SUM(cen.Age_40to44) + SUM(cen.Age_45to49) + SUM(cen.Age_50to54) AS age_25to54_cnt,
        SUM(Age_all)
            - ( --child count
                SUM(cen.Age_under5) + SUM(cen.Age_5to9) + SUM(cen.Age_10to14) + SUM(cen.Age_15to19)
                )
            - ( --senior citizen count
                SUM(cen.Age_over84) + SUM(cen.Age_80to84) + SUM(cen.Age_75to79) + SUM(cen.Age_70to74)
                )
            - ( --young adult count
                SUM(cen.Age_20to24) + SUM(cen.Age_25to29) + SUM(cen.Age_30to34)
                )
            - ( --prime earning years count
                SUM(cen.Age_35to39) + SUM(cen.Age_40to44) + SUM(cen.Age_45to49) + SUM(cen.Age_50to54)
                )
         AS age_55to70_cnt,
    /* ethnicity */
         SUM(cen.Ethnicity_White) AS eth_euro_cnt,
         SUM(cen.Ethnicity_AfricanAmerican) AS eth_african_cnt,
         SUM(cen.Ethnicity_Asian) + SUM(cen.Ethnicity_PacificIslander) AS eth_asiapac_cnt,
         SUM(cen.Ethnicity_NativeAmerican) + SUM(cen.Ethnicity_Other) AS eth_other_cnt,
         SUM(cen.Ethnicity_All_HispancLatino_Descent) AS eth_hislat_descent_cnt,
    /* total pop */
        SUM(cen.Age_all) as total_pop_cnt
    FROM
        nyc_census_data cen
    GROUP BY
        cen.zipcode
    ORDER BY
        cen.zipcode ASC
),
```

SUBQUERY PRACTICE #3 – WITH CTE, *PART 2 → CTE FOR IRS DATA*



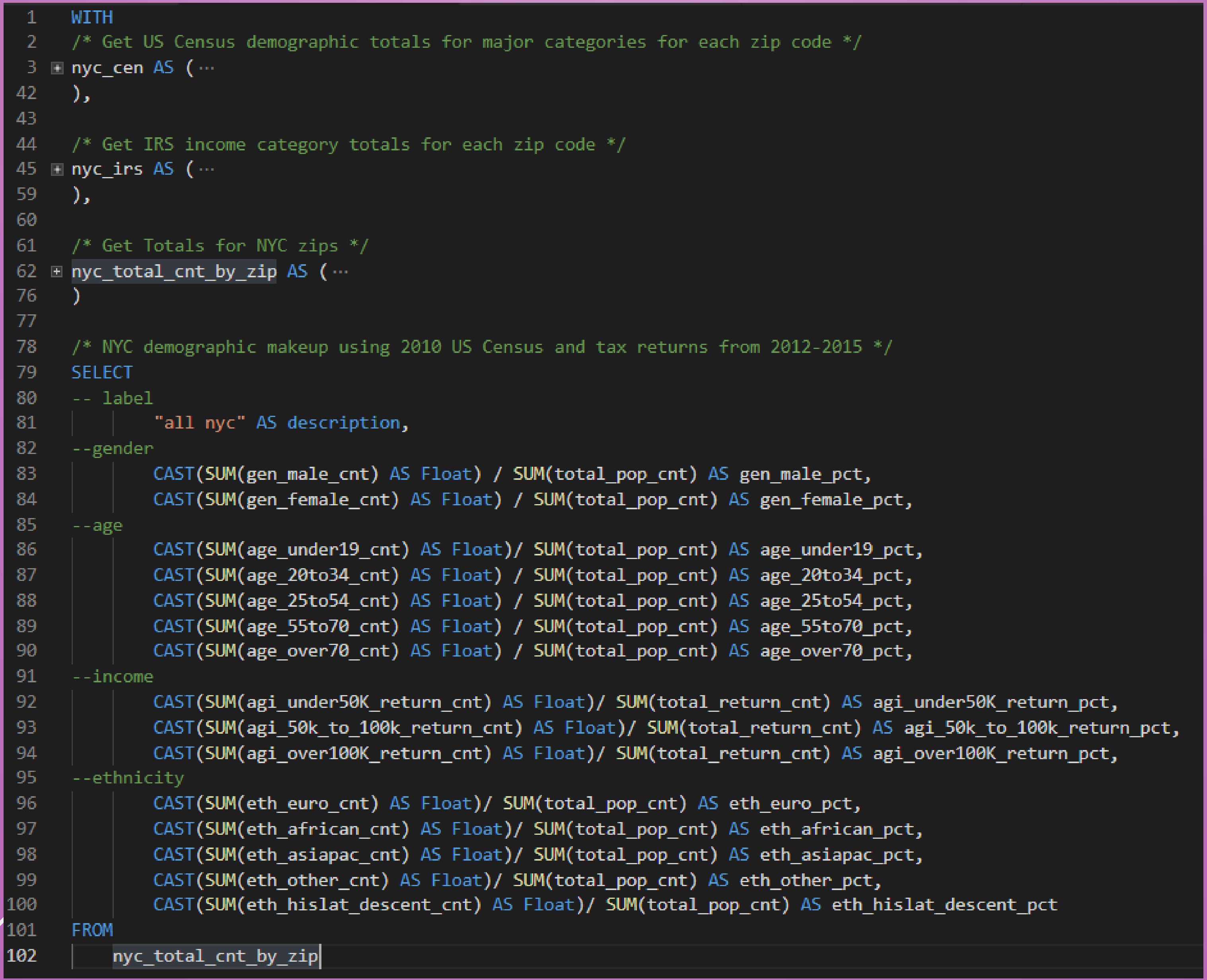
```sql
WITH
/* Get US Census demographic totals for major categories for each zip code */
nyc_cen AS (⋯
),

/* Get IRS income category totals for each zip code */
nyc_irs AS (
    SELECT
        zipcode,
        SUM(CASE WHEN agi_map_id IN (1,2) THEN return_count ELSE NULL END) AS agi_under50k_return_cnt,
        SUM(CASE WHEN agi_map_id IN (3,4) THEN return_count ELSE NULL END) AS agi_50k_to_100k_return_cnt,
        SUM(CASE WHEN agi_map_id IN (5,6) THEN return_count ELSE NULL END) AS agi_over100K_return_cnt,
        SUM(return_count) AS total_return_cnt
    FROM irs_nyc_tax_returns
    WHERE year >= 2012
        AND year <= 2015
    GROUP BY
        zipcode
    ORDER BY
        zipcode ASC
),
```

SUBQUERY PRACTICE #3 – WITH CTE, PART 3 ➔ CTE FOR COMBINED DATA



```sql
WITH
/* Get US Census demographic totals for major categories for each zip code */
nyc_cen AS (···
),

/* Get IRS income category totals for each zip code */
nyc_irs AS (···
),

/* Get Totals for NYC zips */
nyc_total_cnt_by_zip AS (
    SELECT
        cen.*,
        irs.agi_under50K_return_cnt,
        irs.agi_50k_to_100k_return_cnt,
        irs.agi_over100K_return_cnt,
        irs.total_return_cnt
    FROM
        nyc_cen cen
        INNER JOIN
            nyc_irs irs
                ON (cen.zipcode = irs.zipcode)
    ORDER BY
        cen.zipcode
)
```

SUBQUERY PRACTICE #3 – WITH *CTE, PART 4 → FINAL QUERY*



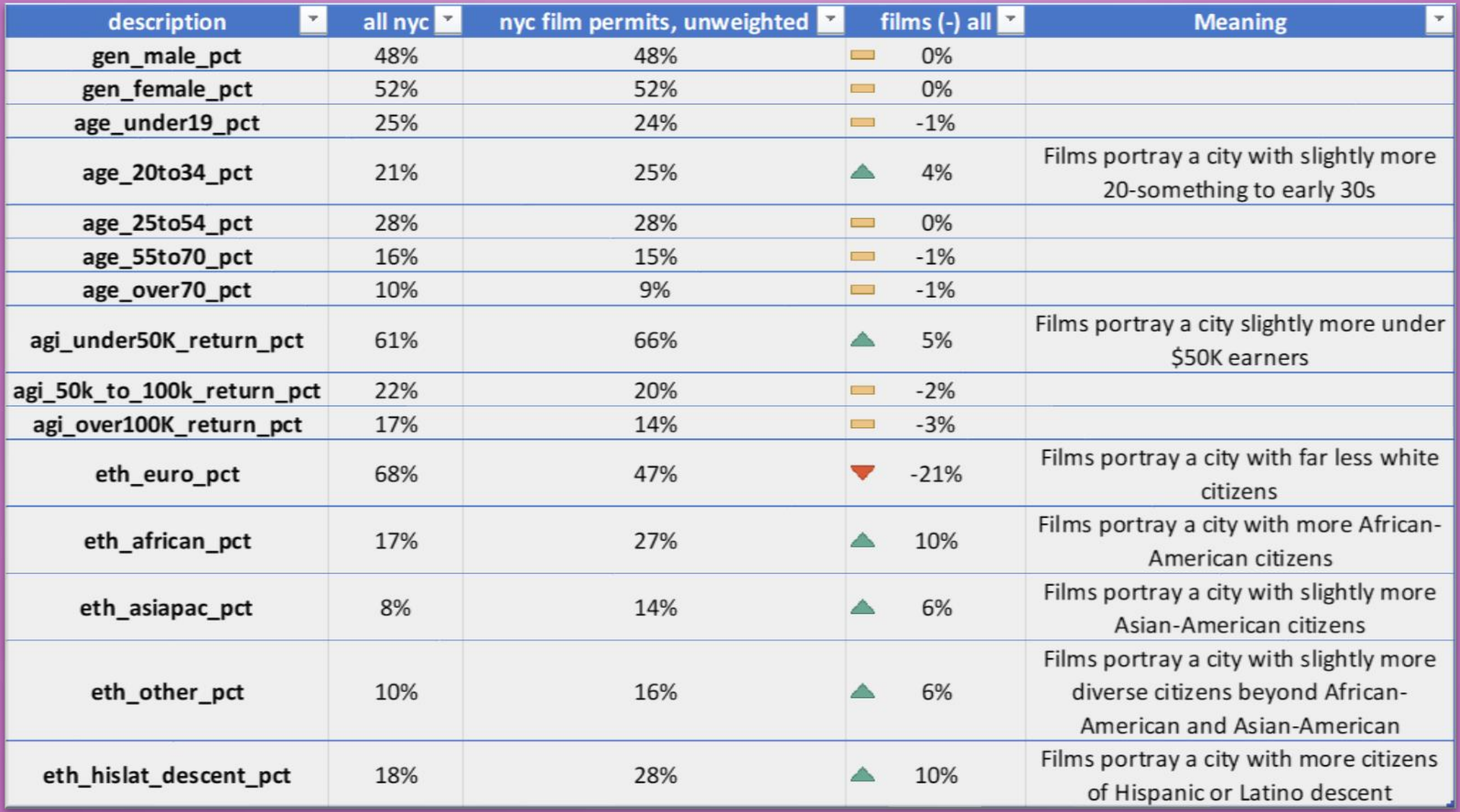
```sql
WITH
/* Get US Census demographic totals for major categories for each zip code */
nyc_cen AS (⋯
),

/* Get IRS income category totals for each zip code */
nyc_irs AS (⋯
),

/* Get Totals for NYC zips */
nyc_total_cnt_by_zip AS (⋯
)

/* NYC demographic makeup using 2010 US Census and tax returns from 2012-2015 */
SELECT
-- label
        "all nyc" AS description,
--gender
        CAST(SUM(gen_male_cnt) AS Float) / SUM(total_pop_cnt) AS gen_male_pct,
        CAST(SUM(gen_female_cnt) AS Float) / SUM(total_pop_cnt) AS gen_female_pct,
--age
        CAST(SUM(age_under19_cnt) AS Float)/ SUM(total_pop_cnt) AS age_under19_pct,
        CAST(SUM(age_20to34_cnt) AS Float) / SUM(total_pop_cnt) AS age_20to34_pct,
        CAST(SUM(age_25to54_cnt) AS Float) / SUM(total_pop_cnt) AS age_25to54_pct,
        CAST(SUM(age_55to70_cnt) AS Float) / SUM(total_pop_cnt) AS age_55to70_pct,
        CAST(SUM(age_over70_cnt) AS Float) / SUM(total_pop_cnt) AS age_over70_pct,
--income
        CAST(SUM(agi_under50K_return_cnt) AS Float)/ SUM(total_return_cnt) AS agi_under50K_return_pct,
        CAST(SUM(agi_50k_to_100k_return_cnt) AS Float)/ SUM(total_return_cnt) AS agi_50k_to_100k_return_pct,
        CAST(SUM(agi_over100K_return_cnt) AS Float)/ SUM(total_return_cnt) AS agi_over100K_return_pct,
--ethnicity
        CAST(SUM(eth_euro_cnt) AS Float)/ SUM(total_pop_cnt) AS eth_euro_pct,
        CAST(SUM(eth_african_cnt) AS Float)/ SUM(total_pop_cnt) AS eth_african_pct,
        CAST(SUM(eth_asiapac_cnt) AS Float)/ SUM(total_pop_cnt) AS eth_asiapac_pct,
        CAST(SUM(eth_other_cnt) AS Float)/ SUM(total_pop_cnt) AS eth_other_pct,
        CAST(SUM(eth_hislat_descent_cnt) AS Float)/ SUM(total_pop_cnt) AS eth_hislat_descent_pct
FROM
    nyc_total_cnt_by_zip
```

SUBQUERY EXAMPLE #4 – WITH CTE

CLASS PROJECT ➔

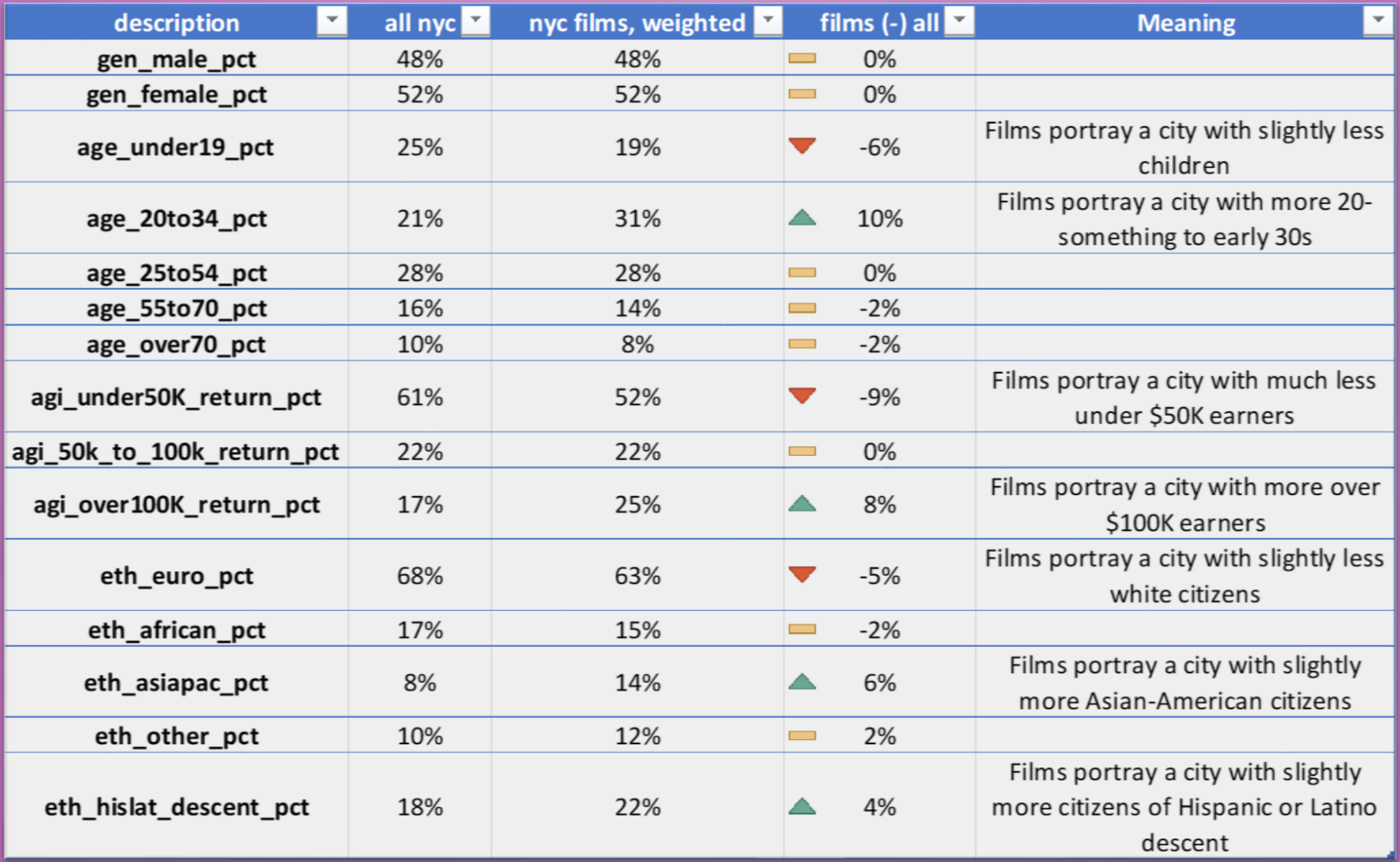
- we want to compare the demographic makeup of NYC versus the demographic makeup of the “NYC on video media (films/TV)”.
- In practice #3 – we found the makeup for all NYC.
- In example #4, we must do the same calculations but only for those zip codes that had filming shoots during 2012-2015.
- We also have to choose whether we weight up/down the zip codes by the number of filming shoots.

SUBQUERY EXAMPLE #4 – WITH CTE *UNWEIGHTED ZIP CODES*



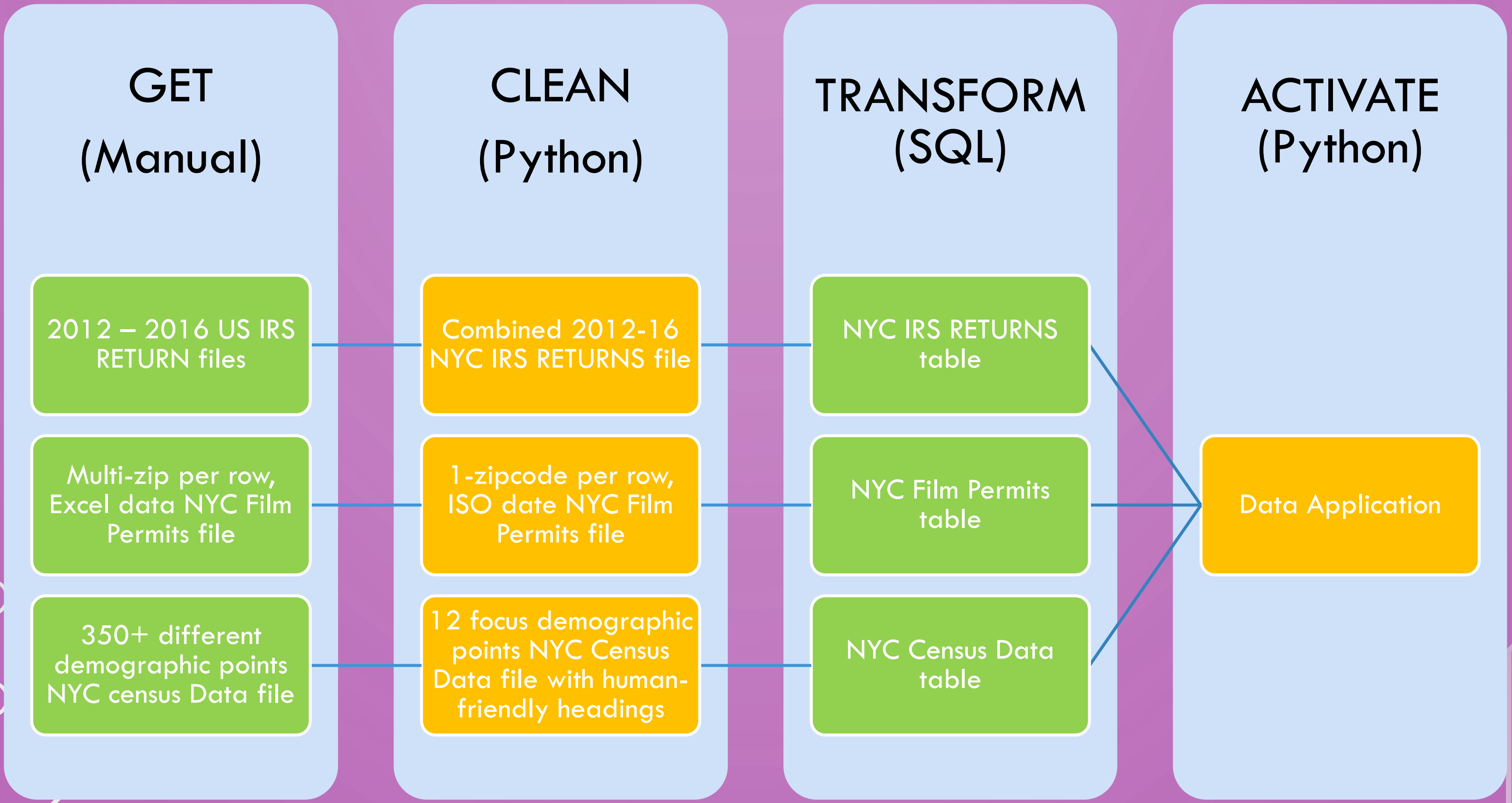
description	all nyc	nyc film permits, unweighted	films (-) all	Meaning
gen_male_pct	48%	48%	▬ 0%	
gen_female_pct	52%	52%	▬ 0%	
age_under19_pct	25%	24%	▬ -1%	
age_20to34_pct	21%	25%	▲ 4%	Films portray a city with slightly more 20-something to early 30s
age_25to54_pct	28%	28%	▬ 0%	
age_55to70_pct	16%	15%	▬ -1%	
age_over70_pct	10%	9%	▬ -1%	
agi_under50K_return_pct	61%	66%	▲ 5%	Films portray a city slightly more under $50K earners
agi_50k_to_100k_return_pct	22%	20%	▬ -2%	
agi_over100K_return_pct	17%	14%	▬ -3%	
eth_euro_pct	68%	47%	▼ -21%	Films portray a city with far less white citizens
eth_african_pct	17%	27%	▲ 10%	Films portray a city with more African-American citizens
eth_asiapac_pct	8%	14%	▲ 6%	Films portray a city with slightly more Asian-American citizens
eth_other_pct	10%	16%	▲ 6%	Films portray a city with slightly more diverse citizens beyond African-American and Asian-American
eth_hislat_descent_pct	18%	28%	▲ 10%	Films portray a city with more citizens of Hispanic or Latino descent

SUBQUERY EXAMPLE #4 – WITH CTE
WEIGHTED ZIP CODES



description	all nyc	nyc films, weighted	films (-) all	Meaning
gen_male_pct	48%	48%	▬ 0%	
gen_female_pct	52%	52%	▬ 0%	
age_under19_pct	25%	19%	▼ -6%	Films portray a city with slightly less children
age_20to34_pct	21%	31%	▲ 10%	Films portray a city with more 20-something to early 30s
age_25to54_pct	28%	28%	▬ 0%	
age_55to70_pct	16%	14%	▬ -2%	
age_over70_pct	10%	8%	▬ -2%	
agi_under50K_return_pct	61%	52%	▼ -9%	Films portray a city with much less under $50K earners
agi_50k_to_100k_return_pct	22%	22%	▬ 0%	
agi_over100K_return_pct	17%	25%	▲ 8%	Films portray a city with more over $100K earners
eth_euro_pct	68%	63%	▼ -5%	Films portray a city with slightly less white citizens
eth_african_pct	17%	15%	▬ -2%	
eth_asiapac_pct	8%	14%	▲ 6%	Films portray a city with slightly more Asian-American citizens
eth_other_pct	10%	12%	▬ 2%	
eth_hislat_descent_pct	18%	22%	▲ 4%	Films portray a city with slightly more citizens of Hispanic or Latino descent

CLASS PROJECT PIPELINE:
NYC MEDIA REPRESENTATION



GET (Manual)	CLEAN (Python)	TRANSFORM (SQL)	ACTIVATE (Python)
2012 – 2016 US IRS RETURN files	Combined 2012-16 NYC IRS RETURNS file	NYC IRS RETURNS table	
Multi-zip per row, Excel data NYC Film Permits file	1-zipcode per row, ISO date NYC Film Permits file	NYC Film Permits table	Data Application
350+ different demographic points NYC census Data file	12 focus demographic points NYC Census Data file with human-friendly headings	NYC Census Data table	

Comparison Operators

Operator	*Description*
=	equals
<>	is not equal to
!=	is not equal to
<	less than
>	greater than
AND	logical and
OR	logical or
NOT	logical not

Other operators

SQL	*Description*
as	used to change the name of a column in the result
distinct	no duplicate rows
order by column(s)	sorts by column(s) in ascending order
order by .. desc	sorts by column(s) in descending order
*	select all columns
like ‘%pattern_’	$: any sequence of characters _: any single character
attribute is null	rows that have null values for the specific attribute
is not null	rows that have not null values for the specific attribute
between this and that	between **this** value and **that** value
in	set membership
limit n	fetches only the top n rows from the database