SQL Data Analysis

Student Data Base

In this SQL project we used the Students Performance dataset from Kaggle you can refer it from this link [Student](). Big thank to Mr Joakim Arvidsson.

Database creation

We will start by creating the schema:

```
create schema student_schema;
set search_path = student_schema;
```

Then we will move to create tables :

```
create table education_habit(
        student_id varchar(20) primary key,
        weekly_study_hours integer ,
        non_scientific_books_reading_frequeny varchar(15),
        scientific_books_reading_frequeny varchar(15),
        department_conferences_attendancy varchar(5),
        projects_activities_impacts varchar(15),
        class_attendence varchar(15),
        mid_term_exams_preparation varchar(15),
        mid_term_exams_preparation_time varchar(50),
        taking_notes_in_class varchar(15),
        listening_in_class varchar(15),
        course_discussion varchar(15),
        flip_classroom varchar(20),
        avg_cum_grade_last_semester float,
        avg_expected_cum_grade_graduation float,
        course_id integer,
        output_grade varchar(20)
        )
```

```
create table family_question(
        student_id varchar(20) primary key,
        mother_education varchar(20),
        father_education varchar(20),
        nb_sisters_brothers integer,
        parental_status varchar(30),
        mother_occupation varchar(30),
        father_ocupation varchar(30)
        )

create table personal_question(
        student_id varchar(20) primary key,
        student_age integer,
        student_gender varchar(10),
        high_school_type_graduated varchar(10),
        scholarship_type integer,
        additional_work varchar(5),
        regular_artistic_or_sports_activity varchar(5),
        is_there_partner varchar(5),
        total_salary integer,
        university_transportation varchar(20),
        cyprus_accommodation_type varchar(15)
        )
```

We use those queries to display all table content:

```
select * from personal_question
select * from education_habit
select * from family_question
```

Basic Queries:

1) Retrieve all students who have a GPA greater than 3.00

```
select student_id , avg_cum_grade_last_semester
from education_habit
```

where avg_cum_grade_last_semester > 3.00

	student_id [PK] character varying (20)	avg_cum_grade_last_semester double precision
1	STUDENT6	3.37
2	STUDENT7	3.44
3	STUDENT9	3.2
4	STUDENT12	3.41
5	STUDENT13	3.48
6	STUDENT14	3.39
7	STUDENT15	3.31
8	STUDENT17	3.33
9	STUDENT21	3.02
10	STUDENT25	3.13
11	STUDENT29	3.54
12	STUDENT30	3.39
13	STUDENT31	3.98
14	STUDENT32	3.04
15	STUDENT35	3.03
16	STUDENT37	3.35
17	STUDENT38	3.58
18	STUDENT39	3.32

Total rows: 65 of 65 Query complete 00:00:00.317

2) List all students who have a full scholarship

select student_id, scholarship_type as full_scholarship

from personal_question

where scholarship_type = 100

	student_id [PK] character varying (20)	full_scholarship integer
1	STUDENT14	100
2	STUDENT17	100
3	STUDENT21	100
4	STUDENT22	100
5	STUDENT31	100
6	STUDENT39	100
7	STUDENT65	100
8	STUDENT86	100
9	STUDENT92	100
10	STUDENT96	100
11	STUDENT109	100
12	STUDENT112	100
13	STUDENT115	100
14	STUDENT116	100
15	STUDENT117	100
16	STUDENT119	100
17	STUDENT123	100
18	STUDENT126	100

Total rows: 23 of 23 Query complete 00:00:00.51

3) Find students who are engaged in both additional work and regular artistic or sports activity?

```
select student_id, additional_work, regular_artistic_or_sports_activity
from personal_question
where additional_work='yes' and regular_artistic_or_sports_activity='yes'
```

	student_id [PK] character varying (20)	additional_work character varying (5)	regular_artistic_or_sports_activity character varying (5)
1	STUDENT8	yes	yes
2	STUDENT12	yes	yes
3	STUDENT15	yes	yes
4	STUDENT24	yes	yes
5	STUDENT31	yes	yes
6	STUDENT38	yes	yes
7	STUDENT52	yes	yes
8	STUDENT58	yes	yes
9	STUDENT68	yes	yes
10	STUDENT79	yes	yes
11	STUDENT92	yes	yes
12	STUDENT96	yes	yes
13	STUDENT106	yes	yes
14	STUDENT108	yes	yes
15	STUDENT115	yes	yes
16	STUDENT121	yes	yes
17	STUDENT122	yes	yes
18	STUDENT125	yes	yes

Total rows: 27 of 27 Query complete 00:00:00.266

Aggregation and Grouping:

4) Find the average GPA for each age group.

```
select student_age ,avg(avg_cum_grade_last_semester ) as average_gpa
from  education_habit
join personal_question on education_habit.student_id = personal_question.student_id
group by student_age
order by average_gpa desc
```

	student_age integer	average_gpa double precision
1	43	3.87
2	42	3.8
3	27	3.54
4	33	3.53
5	34	3.45
6	32	3.31
7	26	3.18
8	20	3.099333333333334
9	24	3.047142857142857
10	22	2.8564999999999996
11	41	2.63
12	19	2.605
13	21	2.579090909090909
14	18	2.5749999999999997
15	23	2.5
16	25	2.3570588235294117
17	44	2.19

Total rows: 17 of 17 Query complete 00:

5) Calculate the percentage of students in each transportation category.

```
select university_transportation,count(student_id)*100.0/(select count(student_id)from
personal_question) as percentage
from personal_question
group by university_transportation
```

	university_transportation character varying (20)	percentage numeric
1	Other	14.4827586206896552
2	bicycle	0.68965517241379310345
3	Private car/taxi	17.2413793103448276
4	Bus	67.5862068965517241

6) Determine the most common father's occupation among students with a GPA above 3.5

```
select count(e.student_id), f.father_ocupation
from family_question f
join education_habit e on  f.student_id = e.student_id
where e.avg_cum_grade_last_semester >3.5
group by f.father_ocupation
```

	student_id [PK] character varying (20)	highest_gpa double precision
1	STUDENT31	3.98
2	STUDENT95	3.93
3	STUDENT48	3.92
4	STUDENT92	3.91
5	STUDENT142	3.88
6	STUDENT67	3.88
7	STUDENT82	3.87
8	STUDENT85	3.86
9	STUDENT58	3.82
10	STUDENT56	3.8

Filtering and Sorting:

7) List the top 10 students with the highest GPA

```
select student_id, max(avg_cum_grade_last_semester) as highest_GPA
from education_habit
group by avg_cum_grade_last_semester ,student_id
order by avg_cum_grade_last_semester desc
limit 10
```

	student_id [PK] character varying (20)	highest_gpa double precision
1	STUDENT31	3.98
2	STUDENT95	3.93
3	STUDENT48	3.92
4	STUDENT92	3.91
5	STUDENT142	3.88
6	STUDENT67	3.88
7	STUDENT82	3.87
8	STUDENT85	3.86
9	STUDENT58	3.82
10	STUDENT56	3.8

8) Find students who have never attended seminars/conferences related to their department

```
select student_id , department_conferences_attendancy
from education_habit
where department_conferences_attendancy = 'no'
```

	student_id [PK] character varying (20)	department_conferences_attendancy character varying (5)
1	STUDENT7	no
2	STUDENT16	no
3	STUDENT18	no
4	STUDENT19	no
5	STUDENT21	no
6	STUDENT74	no
7	STUDENT75	no
8	STUDENT78	no
9	STUDENT103	no
10	STUDENT112	no
11	STUDENT113	no
12	STUDENT114	no
13	STUDENT115	no
14	STUDENT116	no
15	STUDENT117	no
16	STUDENT118	no
17	STUDENT124	no
18	STUDENT125	no

Total rows: 31 of 31 Query complete 00:00:00.359

9) Retrieve students who always attend classes and have a GPA above 3.0

```
select student_id,class_attendence, avg_cum_grade_last_semester
from education_habit
where avg_cum_grade_last_semester >3.0 and class_attendence = 'always'
order by avg_cum_grade_last_semester desc
```

	student_id [PK] character varying (20)	class_attendence character varying (15)	avg_cum_grade_last_semester double precision
1	STUDENT31	always	3.98
2	STUDENT95	always	3.93
3	STUDENT92	always	3.91
4	STUDENT67	always	3.88
5	STUDENT82	always	3.87
6	STUDENT85	always	3.86
7	STUDENT58	always	3.82
8	STUDENT56	always	3.8
9	STUDENT63	always	3.79
10	STUDENT133	always	3.77
11	STUDENT89	always	3.76
12	STUDENT145	always	3.76
13	STUDENT71	always	3.73
14	STUDENT74	always	3.73
15	STUDENT55	always	3.71
16	STUDENT57	always	3.7
17	STUDENT132	always	3.67
18	STUDENT87	always	3.62

Total rows: 55 of 55 Query complete 00:00:00.585

Join Operations:

10) Join student data with course data to find the average grade for each course.

```
select course_id, avg(avg_cum_grade_last_semester)as average_grade
from education_habit
group by course_id
order by average_grade desc
```

	course_id integer	average_grade double precision
1	5	3.141428571428571
2	3	3.0725000000000002
3	2	3.04
4	6	2.9575
5	1	2.854090909090909
6	7	2.771333333333333
7	9	2.598095238095238
8	4	2.21
9	8	2.1599999999999997

11) Find the students who have a partner and their corresponding GPA.

```
select personal_question.student_id, avg_cum_grade_last_semester, is_there_partner
from personal_question join education_habit on personal_question.student_id =
education_habit.student_id
where is_there_partner ='yes'
order by avg_cum_grade_last_semester
```

	student_id character varying (20)	avg_cum_grade_last_semester double precision	is_there_partner character varying (5)
1	STUDENT139	0.17	yes
2	STUDENT122	0.43	yes
3	STUDENT78	0.99	yes
4	STUDENT8	1.18	yes
5	STUDENT10	1.21	yes
6	STUDENT140	1.48	yes
7	STUDENT26	1.85	yes
8	STUDENT97	2.01	yes
9	STUDENT116	2.05	yes
10	STUDENT27	2.13	yes
11	STUDENT34	2.14	yes
12	STUDENT102	2.23	yes
13	STUDENT40	2.24	yes
14	STUDENT49	2.3	yes
15	STUDENT121	2.3	yes
16	STUDENT127	2.31	yes
17	STUDENT137	2.37	yes
18	STUDENT20	2.37	yes

Total rows: 61 of 61 Query complete 00:00:00.310

12) List all students along with their mother's and father's education levels.

```
select student_id, mother_education,father_education
from family_question
```

	student_id [PK] character varying (20)	mother_education character varying (20)	father_education character varying (20)
1	STUDENT1	primary school	secondary school
2	STUDENT2	secondary school	high school
3	STUDENT3	secondary school	secondary school
4	STUDENT4	primary school	secondary school
5	STUDENT5	high school	high school
6	STUDENT6	high school	high school
7	STUDENT7	primary school	high school
8	STUDENT8	university	high school
9	STUDENT9	secondary school	university
10	STUDENT10	primary school	secondary school
11	STUDENT11	high school	university
12	STUDENT12	MSc	MSc
13	STUDENT13	high school	MSc
14	STUDENT14	secondary school	secondary school
15	STUDENT15	high school	primary school
16	STUDENT16	university	university
17	STUDENT17	secondary school	secondary school
18	STUDENT18	secondary school	secondary school

Total rows: 145 of 145 Query complete 00:00:00.437

Subqueries:

13) Find students who have a higher GPA than the average GPA.

```
select student_id, avg_cum_grade_last_semester
from education_habit
where avg_cum_grade_last_semester> (select avg(avg_cum_grade_last_semester)from
education_habit)
order by avg_cum_grade_last_semester desc
```

	student_id [PK] character varying (20)	avg_cum_grade_last_semester double precision
1	STUDENT31	3.98
2	STUDENT95	3.93
3	STUDENT48	3.92
4	STUDENT92	3.91
5	STUDENT142	3.88
6	STUDENT67	3.88
7	STUDENT82	3.87
8	STUDENT85	3.86
9	STUDENT58	3.82
10	STUDENT56	3.8
11	STUDENT63	3.79
12	STUDENT133	3.77
13	STUDENT145	3.76
14	STUDENT89	3.76
15	STUDENT74	3.73
16	STUDENT71	3.73
17	STUDENT55	3.71
18	STUDENT57	3.7

Total rows: 80 of 80 Query complete 00:00:00.574

14) List students who have a salary in the highest salary category.

```
select student_id, total_salary
from personal_question
where total_salary > 410
order by total_salary desc
```

	student_id [PK] character varying (20)	total_salary integer
1	STUDENT104	991
2	STUDENT61	795
3	STUDENT24	707
4	STUDENT139	695
5	STUDENT144	630

15) Retrieve students who prepared for midterm exams regularly during the semester and have a GPA above 3.5.

```
select student_id, avg_cum_grade_last_semester, mid_term_exams_preparation_time
from education_habit
where mid_term_exams_preparation_time = 'regularly during the semester' and
avg_cum_grade_last_semester>3.5
order by avg_cum_grade_last_semester
```

	student_id [PK] character varying (20)	avg_cum_grade_last_semester double precision	mid_term_exams_preparation_time character varying (50)
1	STUDENT132	3.67	regularly during the semester
2	STUDENT57	3.7	regularly during the semester
3	STUDENT55	3.71	regularly during the semester
4	STUDENT74	3.73	regularly during the semester
5	STUDENT89	3.76	regularly during the semester

Index Optimization:

16) Create an index on the Cumulative grade point average column and compare the query performance for retrieving students based on GPA.

Without index:

```
explain analyse
select student_id from education_habit
where avg_cum_grade_last_semester>3.00
```

	QUERY PLAN text
1	Seq Scan on education_habit (cost=0.00..5.81 rows=65 width=10) (actual time=0.077..0.191 rows=65 loops...
2	Filter: (avg_cum_grade_last_semester > '3'::double precision)
3	Rows Removed by Filter: 80
4	Planning Time: 3.516 ms
5	Execution Time: 0.939 ms

With index:

```
create index avg_grade on education_habit(avg_cum_grade_last_semester,student_id)
explain analyse
select student_id from education_habit
where avg_cum_grade_last_semester>3.00
```

	QUERY PLAN text
1	Seq Scan on education_habit (cost=0.00..5.81 rows=65 width=10) (actual time=0.051..0.152 rows=65 loops...
2	Filter: (avg_cum_grade_last_semester > '3'::double precision)
3	Rows Removed by Filter: 80
4	Planning Time: 4.058 ms
5	Execution Time: 0.197 ms

17) Index the Student Age and Sex columns and compare the performance of a query that filters by these columns before and after indexing.

Without index

```
explain analyse
select *
from personal_question
where student_age = 23 and student_gender = 'Female'
```

	QUERY PLAN text
1	Seq Scan on personal_question (cost=0.00..4.17 rows=5 width=56) (actual time=0.055..0.095 rows=4 loops...
2	Filter: ((student_age = 23) AND ((student_gender)::text = 'Female'::text))
3	Rows Removed by Filter: 141
4	Planning Time: 0.246 ms
5	Execution Time: 0.142 ms

With index

```
create index age_sex on personal_question(student_age, student_gender)
explain analyse
select *
from personal_question
where student_age = 23 and student_gender = 'Female'
```

	QUERY PLAN text
1	Seq Scan on personal_question (cost=0.00..4.17 rows=5 width=56) (actual time=0.042..0.069 rows=4 loops...
2	Filter: ((student_age = 23) AND ((student_gender)::text = 'Female'::text))
3	Rows Removed by Filter: 141
4	Planning Time: 0.194 ms
5	Execution Time: 0.093 ms

18) Test the performance of a join query before and after indexing the foreign key columns.

Without index

```
explain analyse
select *
from education_habit eh join personal_question pq
```

on eh.student_id = pq.student_id

	QUERY PLAN text
1	Hash Join (cost=5.26..11.10 rows=145 width=198) (actual time=0.340..0.662 rows=145 loops=1)
2	Hash Cond: ((eh.student_id)::text = (pq.student_id)::text)
3	-> Seq Scan on education_habit eh (cost=0.00..5.45 rows=145 width=142) (actual time=0.070..0.119 rows=145 loops=1)
4	-> Hash (cost=3.45..3.45 rows=145 width=56) (actual time=0.230..0.231 rows=145 loops=1)
5	Buckets: 1024 Batches: 1 Memory Usage: 22kB
6	-> Seq Scan on personal_question pq (cost=0.00..3.45 rows=145 width=56) (actual time=0.032..0.069 rows=145 loops=1)
7	Planning Time: 2.347 ms
8	Execution Time: 0.952 ms

With index

```
create index education_index on education_habit(student_id);
create index personal_index on personal_question(student_id);
```

```
explain analyse
select *
from education_habit eh join personal_question pq
on eh.student_id = pq.student_id;
```

	QUERY PLAN text
1	Hash Join (cost=5.26..11.10 rows=145 width=198) (actual time=0.196..0.503 rows=145 loops=1)
2	Hash Cond: ((eh.student_id)::text = (pq.student_id)::text)
3	-> Seq Scan on education_habit eh (cost=0.00..5.45 rows=145 width=142) (actual time=0.040..0.081 rows=145 loops=1)
4	-> Hash (cost=3.45..3.45 rows=145 width=56) (actual time=0.133..0.134 rows=145 loops=1)
5	Buckets: 1024 Batches: 1 Memory Usage: 22kB
6	-> Seq Scan on personal_question pq (cost=0.00..3.45 rows=145 width=56) (actual time=0.026..0.059 rows=145 loops=1)
7	Planning Time: 0.913 ms
8	Execution Time: 0.592 ms

Complex Queries:

19) Find the correlation between the type of high school graduated from and the students' final grades.

```
select corr(weekly_study_hours::int, avg_cum_grade_last_semester::int) as correlation
from personal_question pq
join education_habit eh on pq.student_id = eh.student_id
```

	correlation double precision
1	-0.07368691729911225

20) Analyze the impact of parental status on the students' GPA.

```
select fq.parental_status, round(avg(eh.avg_cum_grade_last_semester)::numeric,2) as
average_gpa
from family_question fq
join education_habit eh on fq.student_id = eh.student_id
group by (parental_status)
```

	parental_status character varying (30)	average_gpa numeric
1	divorced	2.72
2	married	2.75
3	died_one of them or both	2.92

21) Determine the relationship between weekly study hours and the expected cumulative GPA at graduation.

To determine the relationship we can use the avg:

```
select weekly_study_hours, avg(avg_expected_cum_grade_graduation) as avg_gpa
from education_habit
group by weekly_study_hours
order by avg_gpa desc
```

	weekly_study_hours integer	avg_gpa double precision
1	16	3.06
2	21	2.88
3	13	2.88
4	11	2.87
5	29	2.87
6	10	2.8445454545454547
7	14	2.83
8	8	2.7
9	3	2.6783333333333332
10	6	2.6700000000000004
11	19	2.63
12	2	2.5200000000000005
13	4	2.51875
14	35	2.49
15	1	2.4511764705882353
16	0	2.3634482758620696
17	7	2.3516666666666666
18	15	2.325

Total rows: 20 of 20 Query complete 00:00:00.8

Also, we can use the correlation:

```
select corr(weekly_study_hours::int ,avg_expected_cum_grade_graduation::int) as correlation
from education_habit
```

	correlation double precision
1	0.04336349044067117

22) Create a view to show students' GPA and their parents' education levels.

```
create view students_gpa_parents_education as
select eh.avg_cum_grade_last_semester, fq.mother_education, fq.father_education
from education_habit eh
join family_question fq on eh.student_id = fq.student_id
order by avg_cum_grade_last_semester desc
```

```
select * from students_gpa_parents_education
```

	avg_cum_grade_last_semester double precision	mother_education character varying (20)	father_education character varying (20)
1	0.17	high school	primary school
2	0.43	primary school	primary school
3	0.81	high school	secondary school
4	0.81	primary school	primary school
5	0.85	university	university
6	0.92	university	university
7	0.99	Ph.D	MSc
8	1.18	university	high school
9	1.21	primary school	secondary school
10	1.34	primary school	primary school
11	1.41	primary school	secondary school
12	1.48	high school	high school
13	1.5	high school	university
14	1.84	high school	university
15	1.85	primary school	university
16	1.9	primary school	secondary school
17	1.93	university	university
18	2.01	secondary school	secondary school

Total rows: 145 of 145 Query complete 00:00:00.219

23) Create a view to list students with full scholarships and their average weekly study hours.

```
create view full_scholarships_students_study_hours as
select eh.student_id , pq.scholarship_type, eh.weekly_study_hours
```

```
from education_habit eh join personal_question pq
on eh.student_id = pq.student_id
where pq.scholarship_type = 100
order by (eh.weekly_study_hours)
```

```
select * from full_scholarships_students_study_hours
```

	student_id character varying (20)	scholarship_type integer	weekly_study_hours integer
1	STUDENT14	100	0
2	STUDENT65	100	0
3	STUDENT86	100	0
4	STUDENT112	100	0
5	STUDENT119	100	0
6	STUDENT17	100	1
7	STUDENT96	100	1
8	STUDENT138	100	1
9	STUDENT115	100	2
10	STUDENT92	100	2
11	STUDENT31	100	3
12	STUDENT123	100	3
13	STUDENT133	100	4
14	STUDENT39	100	6
15	STUDENT134	100	6
16	STUDENT116	100	6
17	STUDENT22	100	7
18	STUDENT109	100	7

Total rows: 23 of 23 Query complete 00:00:00.279

24) Create a composite index on transportation and accommodation type, and test the query performance.

Without index

```
explain analyse
select university_transportation, cyprus_accommodation_type
from personal_question
```

	QUERY PLAN text
1	Seq Scan on personal_question (cost=0.00..3.45 rows=145 width=14) (actual time=0.345..0.425 rows=145 loops...
2	Planning Time: 0.251 ms
3	Execution Time: 0.475 ms

With index

```
create index on personal_question (university_transportation, cyprus_accommodation_type)
explain analyse
select university_transportation, cyprus_accommodation_type
from personal_question
```

	QUERY PLAN text
1	Seq Scan on personal_question (cost=0.00..3.45 rows=145 width=14) (actual time=0.071..0.251 rows=145 loops...
2	Planning Time: 4.990 ms
3	Execution Time: 0.327 ms

Views:

25) Create a view that shows the average GPA by weekly study hours.

```
create view avg_gpa_weekly_study as
select weekly_study_hours, round(avg(avg_cum_grade_last_semester)::int,2)as avg_gpa
from education_habit
group by weekly_study_hours

select * from avg_gpa_weekly_study
```

	weekly_study_hours integer	avg_gpa numeric
1	11	3.00
2	8	4.00
3	19	3.00
4	29	3.00
5	4	3.00
6	21	3.00
7	0	3.00
8	40	0.00
9	14	2.00
10	3	3.00
11	13	2.00
12	10	3.00
13	7	3.00
14	9	3.00
15	35	2.00
16	1	2.00
17	2	3.00

Total rows: 20 of 20 Query complete (

26) Create a view that combines detailed student information from all three tables.

```
create view student_info as
select eh.student_id, eh.avg_cum_grade_last_semester,
eh.avg_expected_cum_grade_graduation,
eh.course_id, eh.output_grade,fq.parental_status, pq.student_age, pq.scholarship_type,
pq.total_salary,pq.university_transportation
from education_habit eh
join family_question fq on eh.student_id = fq.student_id
```

join personal_question pq on eh.student_id = pq.student_id

select * from student_info

	student_id character varying (20)	avg_cum_grade_last_semester double precision	avg_expected_cum_grade_graduation double precision	course_id integer	output_grade character varying (20)	parental_status character varying (30)	student_age integer	scholarship_type integer
1	STUDENT1	1.41	1.5	1	Poor	married	23	50
2	STUDENT2	2.01	2.74	1	Poor	married	22	50
3	STUDENT3	2.49	2.34	1	Poor	married	25	50
4	STUDENT4	2.79	2.27	1	Poor	married	21	50
5	STUDENT5	2.38	2.28	1	Poor	married	22	50
6	STUDENT6	3.37	3.08	1	Below Average	married	24	50
7	STUDENT7	3.44	3.11	1	Good	married	18	75
8	STUDENT8	1.18	1.22	1	Below Average	married	21	50
9	STUDENT9	3.2	2.69	1	Good	married	24	50
10	STUDENT10	1.21	2.26	1	Fail	married	22	50
11	STUDENT11	1.84	1.26	1	Below Average	married	21	50
12	STUDENT12	3.41	2.73	1	Fail	married	21	75
13	STUDENT13	3.48	2	1	Fail	divorced	21	75
14	STUDENT14	3.39	2.46	1	Poor	married	22	100
15	STUDENT15	3.31	3.39	1	Below Average	married	32	75
16	STUDENT16	2.16	2.04	1	Below Average	married	23	50
17	STUDENT17	3.33	2.61	1	Poor	married	19	100

Total rows: 145 of 145 Query complete 00:00:00.329 Ln 265, Col 1

Triggers:

27) Create a trigger to modify the "education_habit" table after each modification (insert, update, delete) applied on the "personal_question" table.

Create the Trigger Function

```
create or replace function education_habit_changes()
returns trigger as $$
begin

    if TG_OP = 'insert' then
        insert into education_habit(
                                    student_id,
                                    weekly_study_hours,
                                    non_scientific_books_reading_frequency,
                                    scientific_books_reading_frequency,
                                    departement_conferences_attendancy,
                                    project_activities_impacts,
                                    class_attendence,
                                    mid_term_exams_preparation,
                                    mid_term_exams_preparation_time,
                                    taking_notes_in_class,
                                    listening_in_class,
                                    course_discussion,
                                    flip_classroom,
                                    avg_cum_grade_last_semester,
                                    avg_expected_cum_grade_graduation,
                                    course_id,output_grade)

                        values (
```

```
                                        New.student_id,
                                        New.weekly_study_hours,
                                        New.non_scientific_books_reading_frequency,

                                                New.scientific_books_reading_frequency,
                                        New.departement_conferences_attendancy,
                                        New.project_activities_impacts,
                                        New.class_attendence,
                                        New.mid_term_exams_preparation,
                                        New.mid_term_exams_preparation_time,
                                        New.taking_notes_in_class,
                                        New.listening_in_class,
                                        New.course_discussion,
                                        New.flip_classroom,
                                        New.avg_cum_grade_last_semester,
                                        New.avg_expected_cum_grade_graduation,
                                        New.course_id,
                                        New.output_grade);
                                        return New;

    elseif TG_OP = 'update'then
        update education_habit
            set
            student_id = New.student_id,
            weekly_study_hours=New.weekly_study_hours ,
            non_scientific_books_reading_frequency=
            New.non_scientific_books_reading_frequency,
            scientific_books_reading_frequency=New.scientific_books_reading_frequency,
            departement_conferences_attendancy=New.departement_conferences_attendancy,
            project_activities_impacts=New.project_activities_impacts,
            class_attendence = New.class_attendence,
            mid_term_exams_preparation = New.mid_term_exams_preparation,
            mid_term_exams_preparation_time = New.mid_term_exams_preparation_time,
            taking_notes_in_class = New.taking_notes_in_class,
            listening_in_class= New.listening_in_class,
            course_discussion= New.course_discussion,
            flip_classroom = New.flip_classroom,
            avg_cum_grade_last_semester = New.avg_cum_grade_last_semester,
            avg_expected_cum_grade_graduation = New.avg_expected_cum_grade_graduation,
            course_id =New.course_id,
            output_grade = New.output_grade
            where student_id = New.student_id;
            return New;



    elseif TG_OP = 'delete' then
        delete from education_habit
                where student_id = Old.student_id;
                return Old;
    end if ;

end;
```

```
$$ language plpgsql;
```

Create Triggers

Trigger for insertion

```
create trigger after_insert_on_personal_question
after insert on personal_question
for each row
execute function education_habit_changes();
```

Trigger for update

```
create trigger after_update_on_personal_question
after update on personal_question
for each row
execute function education_habit_changes();
```

Trigger for delete

```
create trigger after_delete_on_personal_question
after delete on personal_question
for each row
execute function education_habit_changes();
```

Happy Leaning, don't forget to upvote and comment it this makes me happy!!!