KUTAISI
INTERNATIONAL
UNIVERSITY

Lab Assignment November 10th

Fundamentals of Databases 1
Dimitri Tabatadze
20:22 Sunday 12th November, 2023

SporstClub

1.

```sql
SELECT *
FROM member
WHERE parentName IS NOT NULL AND age < 18
```

Result:

memName	isTrainer	DoB	email	postalCode	gender	entryDate	parentName	age
lisa	0	2015-11-19	figaro@xx.ge	4600	f	NULL	figaro	7
val	0	2013-07-12	val@xx.ge	0103	m	2020-05-01	figaro	10
aron	0	2013-11-06	klopp@xx.ge	0103	m	2020-01-01	klopp	10
rose	0	2015-02-10	lion@xx.ge	0107	f	NULL	lion	8
robin	0	2012-09-16	nelly@xx.ge	4600	m	2017-01-01	nelly	11

2.

```sql
SELECT *, COUNT(courseID) AS courseCount
FROM course
GROUP BY area
```

Result:

courseID	courseName	targetGroup	area	trainerName	courseCount
6	high jump	fam	athletics	lena	2
3	running	men	fitness	NULL	4
1	wrestling	men	martialArts	klopp	1
2	waterball	men	watersport	klopp	3

3.

```sql
SELECT *
FROM area a
WHERE 3 <= (
   SELECT COUNT(area)
   FROM course c
   WHERE a.area = c.area
)
```

Result:

area	description	manager
fitness	Comprises all courses that encrease healthy lifestyle and fitness	lena
watersport	All sports that have to do with water	lazy

4.

```sql
SELECT MIN(age), MAX(age) FROM member
```

Result:

MIN(age)	MAX(age)
7	58

5.

```sql
SELECT memName, age, parentName
FROM member
WHERE age = (SELECT MIN(age) FROM member)
```

Result:

memName	age	parentName
lisa	7	figaro

6.

```sql
SELECT *, (
   SELECT COUNT(memName) as amt
   FROM enrollment
   WHERE courseID = c1.courseID
) memberCount
FROM course c1
GROUP BY courseID
```

Result:

courseID	courseName	targetGroup	area	trainerName	memberCount
1	wrestling	men	martialArts	klopp	6
2	waterball	men	watersport	klopp	0
3	running	men	fitness	NULL	0
4	jogging	fam	fitness	lena	6
5	fitnessKids	kid	fitness	klopp	7
6	high jump	fam	athletics	lena	8
7	obstacle race	fam	athletics	lena	6
8	swimming	fam	watersport	NULL	0
9	free style	kid	watersport	lazy	1
10	aerobics	fam	fitness	lena	6

7.

```sql
SELECT COUNT(trainerName) FROM trainer
```

Result:

COUNT(trainerName)
5

8. NULL values get their own group.

```sql
SELECT trainerName , (
   SELECT COUNT(courseID) as amt
   FROM course
   WHERE trainerName = c1.trainerName
) amt
FROM trainer c1
GROUP BY trainerName
```

Result:

trainerName	amt
coach	0
klopp	3
lazy	1
lena	4
nelly	0

9.

```sql
SELECT *, TIMESTAMPDIFF(YEAR , DoB , entryDate) AS entryAge
FROM member
```

Result:

memName	isTrainer	DoB	email	postalCode	gender	entryDate	parentName	age	entryAge
aron	0	2013-11-06	klopp@xx.ge	0103	m	2020-01-01	klopp	10	6
coach	1	1988-12-01	coach@xx.ge	4565	m	NULL	NULL	34	NULL
figaro	0	1989-05-07	figaro@xx.ge	3674	m	NULL	NULL	34	NULL
hope	0	1965-09-14	hope@xx.ge	4565	f	2016-03-01	NULL	58	50
klopp	1	1980-12-24	klopp@xx.ge	4600	m	NULL	NULL	42	NULL
lazy	1	1996-11-25	lazy@xx.ge	4600	m	NULL	NULL	26	NULL
lena	1	1995-01-25	lena@xx.ge	0105	f	NULL	NULL	28	NULL
lion	0	1990-10-10	lion@xxx.ge	0103	m	NULL	NULL	33	NULL
lisa	0	2015-11-19	figaro@xx.ge	4600	f	NULL	figaro	7	NULL
luke	0	1998-11-22	luke@xxx.ge	4565	m	NULL	hope	24	NULL
nelly	1	1990-04-21	luke@xx.ge	4565	f	NULL	NULL	33	NULL
robin	0	2012-09-16	nelly@xx.ge	4600	m	2017-01-01	nelly	11	4
rose	0	2015-02-10	lion@xx.ge	0107	f	NULL	lion	8	NULL
val	0	2013-07-12	val@xx.ge	0103	m	2020-05-01	figaro	10	6
valerie	0	1970-03-20	val@xx.ge	0105	f	NULL	NULL	53	NULL

10.

```
SELECT AVG(TIMESTAMPDIFF(YEAR, entryDate, CURDATE())) AS avgEntryYear
FROM member
```

Result:

avgEntryYear
4.7500

11.

```
SELECT COUNT(memName), (
   CASE
      WHEN 0 <= age AND age <= 9 THEN "kid"
      WHEN 10 <= age AND age <= 19 THEN "teen"
      WHEN 20 <= age AND age <= 29 THEN "youngAdult"
      WHEN 30 <= age AND age <= 39 THEN "familiyages"
      WHEN 40 <= age AND age <= 49 THEN "professionalAgers"
      WHEN 50 = age AND age <= 59 THEN "bestAgers"
      ELSE "other"
   END
) AS ageGroup
FROM member
GROUP BY ageGroup
```

Result:

COUNT(memName)	ageGroup
4	familiyages
2	kid
2	other
1	professionalAgers
3	teen
3	youngAdult

12.

```
SELECT COUNT(memName), (
   CASE
      WHEN 0 <= age AND age <= 9 THEN "kid"
      WHEN 10 <= age AND age <= 19 THEN "teen"
      WHEN 20 <= age AND age <= 29 THEN "youngAdult"
      WHEN 30 <= age AND age <= 39 THEN "familiyages"
      WHEN 40 <= age AND age <= 49 THEN "professionalAgers"
      WHEN 50 = age AND age <= 59 THEN "bestAgers"
      ELSE "other"
   END
) AS ageGroup
FROM member
GROUP BY ageGroup
ORDER BY age asc
```

Result:

COUNT(memName)	ageGroup
2	kid
3	teen
3	youngAdult
4	familiyages
1	professionalAgers
2	other

University

13.

```
SELECT profID, (
   SELECT SUM(contactHours)
   FROM course
   WHERE profID = p.profID
) contactHours
FROM professor p
```

Result:

profID	contactHours
2125	10
2126	8
2127	NULL
2133	2
2134	2
2136	NULL
2137	8

14.

```
SELECT COUNT(studID)
FROM (
   SELECT studID
   FROM enrollment
   GROUP BY studID
) X
```

Result:

COUNT(studID)
6

15.

```
SELECT studID, name, semester
FROM student
WHERE studID in (
   SELECT studID
   FROM enrollment
   GROUP BY studID
)
```

Result:

studID	name	semester
25403	Jonas	12
26120	Fichte	10
27550	Schopenhauer	0
28106	Carnap	3
29120	Theophrastos	0
29555	Feuerbach	2

16.

```
SELECT studID, name, semester
FROM student s
WHERE 1 = (
   SELECT COUNT(studID)
   FROM enrollment
   WHERE studID = s.studID
)
```

Result:

studID	name	semester
25403	Jonas	12
26120	Fichte	10

17.

```
SELECT profID, name, rank
FROM professor p
WHERE 1 < (
   SELECT COUNT(assistantID)
   FROM assistant
   WHERE p.profID = profID
)
```

Result:

profID	name	rank
2125	Sokrates	C4
2127	Kopernikus	C3

The results for each query were auto-generated at the compile-time of the pdf file you are viewing :)