

به نام آفریننده بیت‌ها



دانشکده‌ی برق و کامپیوتر
دانشگاه صنعتی اصفهان
نیم‌سال اول ۱۴۰۱ - ۱۴۰۲

پایگاه داده ۱

تمرین سوم

نام:
دانیال خراسانی‌زاده
شماره دانشجویی:
۹۹۲۲۳۹۳
استاد درس:
دکتر بصیری



فهرست مطالب

۱	۱
۲	۲
۳	۳
۴	۴
۵	۵
۶	۶
۷	۷
۸	۸
۹	۹
۱۰	۱۰



۱

حالت بیشینه زمانی رخ می‌دهد که یک کاربر ۱۰۰ شماره تلفن داشته باشد و برای بقیه کاربرها شماره تلفنی ثبت نشده باشد و حالت کمینه زمانی رخ می‌دهد که هر کاربر یک شماره تلفن داشته باشد.

	Minimum	Maximum
Inner Join	100	100
Left outer join	100	199
Right outer join	100	100
Full outer join	100	199

۲

۱.۲

این کوئری تمام گره‌هایی که حداقل یک یال به آن‌ها وارد شده اما هیچ یالی از آن‌ها خارج نشده است را باز می‌گرداند.

۲.۲

ساب کوئری اول گره‌های حاضر به همراه تعداد یال‌های خروجی از آن‌ها را پیدا می‌کند و ساب کوئری دوم گره‌های حاضر به همراه تعداد یال‌های ورودی به آن‌ها را باز می‌گرداند. با جوین این دو کوئری با شروط داده شده در نهایت لیست گره‌هایی که تعداد یال‌های خروجی از آن‌ها از تعداد یال‌های ورودی به آن‌ها کمتر است را در نتیجه خواهیم داشت.



```
1 CREATE VIEW student_instructor AS
2 SELECT student.id,
3        student.name,
4        CASE
5            WHEN student.dept_name LIKE '%Eng.' THEN 'Engineer'::character(9)
6            ELSE 'Scientist'::character(9)
7        END AS dept_type,
8        'STU'::character(3) AS person_type
9 FROM student
10 UNION
11 SELECT instructor.id,
12        instructor.name,
13        CASE
14            WHEN instructor.dept_name LIKE '%Eng.' THEN 'Engineer'::character(9)
15            ELSE 'Scientist'::character(9)
16        END AS dept_type,
17        'INS'::character(3) AS person_type
18 FROM instructor;
```

The screenshot shows the pgAdmin 4 interface with the query executed. The 'Data Output' tab is active, displaying the results of the query. The results are as follows:

id	name	dept_type	person_type
1	96153	Scientist	STU
2	37284	Scientist	STU
3	82402	Scientist	STU
4	41596	Scientist	STU
5	58413	Scientist	STU
6	19603	Scientist	STU
7	48660	Scientist	STU
8	9933	Scientist	STU
9	51416	Scientist	STU
10	94697	Scientist	STU
11	32119	Engineer	STU
12	49073	Scientist	STU
13	61356	Scientist	STU

Total rows: 1000 of 2050 Query complete 00:00:00.113



```
1 SELECT student_instructor.name,
2     student_instructor.person_type,
3     instructor.salary / department.budget AS calc_number
4 FROM department,
5     instructor,
6     student_instructor
7 WHERE person_type = 'INS'
8     AND instructor.id = student_instructor.id
9     AND instructor.dept_name = department.dept_name
10 UNION ALL
11 SELECT student_instructor.name,
12     student_instructor.person_type,
13     department.budget / (
14         SELECT COUNT(*)
15         FROM student AS S
16         WHERE S.dept_name = T.dept_name
17     ) AS calc_number
18 FROM department,
19     student AS T,
20     student_instructor
21 WHERE student_instructor.person_type = 'STU'
22     AND T.id = student_instructor.id
23     AND T.dept_name = department.dept_name;
```

name	person_type	calc_number
1 Lilley	INS	0.15116276246350308
2 McKinnon	INS	0.11872758660070426
3 Levine	INS	0.32476261592829736
4 Yazdi	INS	0.1338691120980485
5 Voronina	INS	0.12857862265751197
6 Dussierre	INS	0.314029387794324
7 Valtchev	INS	0.11895448584029399
8 Desyl	INS	0.08116532697715355
9 Romero	INS	0.12809975745152796
10 Gustafsson	INS	0.2984670138363399
11 Wieland	INS	0.21729921872377157
12 Ullman	INS	0.10706817286185263
13 Choi	INS	0.1463789618899034

Total rows: 1000 of 2050 Query complete 00:00:00.561



```
1 ALTER TABLE film
2 ADD CHECK (length >= 50) NOT VALID;
3 ALTER TABLE payment
4 ADD pay_type VARCHAR(11) CHECK (pay_type in ('credit_card', 'cash', 'online'));
```



```
1 BEGIN;
2 INSERT INTO department
3 VALUES ('Medical', 'Pasteur', 700000);
4 INSERT INTO department
5 VALUES ('Dental', 'Pasteur', 800000);
6 COMMIT;
7 BEGIN;
8 UPDATE department
9 SET budget = budget + (
10     SELECT budget * 0.1
11     FROM department
12     WHERE dept_name = 'Medical'
13 )
14 WHERE dept_name = 'Dental';
15 UPDATE department
16 SET budget = budget * 0.9
17 WHERE dept_name = 'Medical';
18 COMMIT;
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'University' database. The main window shows a SQL query being executed. The query is as follows:

```
2 INSERT INTO department
3 VALUES ('Medical', 'Pasteur', 700000);
4 INSERT INTO department
5 VALUES ('Dental', 'Pasteur', 800000);
6 COMMIT;
7 BEGIN;
8 UPDATE department
9 SET budget = budget + (
10     SELECT budget * 0.1
11     FROM department
12     WHERE dept_name = 'Medical'
13 )
14 WHERE dept_name = 'Dental';
15 UPDATE department
16 SET budget = budget * 0.9
17 WHERE dept_name = 'Medical';
18 COMMIT;
```

The 'Data Output' tab shows the results of the query, which is a table with three columns: dept_name, building, and budget. The table contains 22 rows of data.

dept_name	building	budget
Finance	Candlestick	866831.75
Geology	Palmer	406557.93
Cybernetics	Mercer	794541.46
Astronomy	Taylor	617253.94
Athletics	Bronfman	734550.70
Statistics	Taylor	955051.74
Psychology	Thompson	848175.04
Math	Brodhead	777605.11
Elec. Eng.	Main	276527.61
Mech. Eng.	Rauch	520350.65
Dental	Pasteur	870000.00
Medical	Pasteur	630000.00

Total rows: 22 of 22 Query complete 00:00:00.069



```
1 CREATE FUNCTION actor_film(actor_first_name TEXT, actor_last_name TEXT) RETURNS
    TABLE(film_title TEXT, rental_count int) AS $$
2 SELECT (
3     SELECT title
4     FROM film
5     WHERE film.film_id = film_actor.film_id
6 ) AS film_title,
7 (
8     SELECT COUNT(*)
9     FROM rental,
10         inventory
11     WHERE inventory.inventory_id = rental.inventory_id
12         AND film_actor.film_id = inventory.film_id
13 ) AS rental_count
14 FROM film_actor,
15     actor
16 WHERE actor.actor_id = film_actor.actor_id
17     AND actor.first_name = actor_first_name
18     AND actor.last_name = actor_last_name;
19 $$ LANGUAGE SQL;
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for 'DVD Rental'. The main pane shows the SQL query being executed, which is the same as the one in the previous block. The 'Data Output' tab at the bottom shows the results of the function call, which is a table with two columns: 'film_title' (text) and 'rental_count' (integer). The table contains 12 rows of data.

film_title	rental_count
Canyon Stock	19
Dances None	14
Encino Elf	19
Ending Crowds	11
Gandhi Kiwi	17
Godfather Diary	8
Handicap Boon...	28
Honey Ties	19
Horn Working	24
Image Princess	14
Jersey Sassy	6
Loser Hustler	15

Total rows: 25 of 25 Query complete 00:00:00.086 Ln 20, Col 41



```
1 CREATE PROCEDURE change_replacement_cost(film_id_1 INT, film_id_2 INT) LANGUAGE plpgsql AS
    $$
2 DECLARE amount FLOAT;
3 BEGIN
4 SELECT replacement_cost * 0.05 INTO amount
5 FROM film
6 WHERE film_id = film_id_1;
7 UPDATE film
8 SET replacement_cost = replacement_cost - amount
9 WHERE film_id = film_id_1;
10 UPDATE film
11 SET replacement_cost = replacement_cost + amount
12 WHERE film_id = film_id_2;
13 END;
14 $$;
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'DVDrental' database. The main pane shows the SQL query editor with the procedure code. Below the editor, the 'Data Output' tab is active, displaying the results of the query. The results show two rows of film data, with the 'replacement_cost' column highlighted in green, indicating the updated values.

film_id	release_year	language_id	rental_duration	rental_rate	length	replacement_cost	rating	mpaa_rating
1	2006	1	6	0.99	86	19.94	PG	2
2	2006	1	3	4.99	48	14.04	G	2



```
1 ALTER TABLE customer
2 ADD count_check SMALLINT;
3 UPDATE customer
4 SET count_check = 0;
5 CREATE OR Replace FUNCTION bonus_return_date() RETURNS trigger AS $$
6 DECLARE count smallint;
7 BEGIN
8 SELECT count_check INTO count
9 FROM customer
10 WHERE customer_id = NEW.customer_id;
11 IF count = 2 THEN
12 NEW.return_date := NEW.return_date + interval '7 day';
13 UPDATE customer
14 SET count_check = 0
15 WHERE customer_id = NEW.customer_id;
16 ELSE
17 UPDATE customer
18 SET count_check = count_check + 1
19 WHERE customer_id = NEW.customer_id;
20 END IF;
21 RETURN NEW;
22 END;
23 $$ LANGUAGE plpgsql;
24 CREATE OR REPLACE TRIGGER check_return_date BEFORE
25 INSERT ON rental FOR EACH ROW EXECUTE FUNCTION bonus_return_date();
```



pgAdmin 4

127.0.0.1:5433/browser/

pgAdmin

File Object Tools Help

Dashboard Properties SQL Statistics Dependencies Dependents Processes DVD Rental/postgres@Postgres

Browsers (1)

Postgres

Databases (3)

DVD Rental

Castes

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

Subscriptions

University

postgres

Login/Group Roles

Tablespaces

Query

Query History

```
1 INSERT INTO rental VALUES (16050, '2022-12-24 20:00:00', 1, 1, '2023-01-01 20:00:00', 1);
2 SELECT * FROM rental ORDER BY rental_id DESC;
```

Data Output Messages Notifications

	rental_id	rental_date	inventory_id	customer_id	return_date	staff_id	last_update
	[PK] integer	timestamp without time zone	integer	smallint	timestamp without time zone	smallint	timestamp without time zone
1	16050	2022-12-24 20:00:00	1	1	2023-01-01 20:00:00	1	2022-12-24 16:44:08.662989
2	16049	2005-08-23 22:50:12	2666	393	2005-08-30 01:01:12	2	2006-02-16 02:30:53
3	16048	2005-08-23 22:43:07	2019	103	2005-08-31 21:33:07	1	2006-02-16 02:30:53
4	16047	2005-08-23 22:42:48	2088	114	2005-08-25 02:48:48	2	2006-02-16 02:30:53
5	16046	2005-08-23 22:26:47	4364	74	2005-08-27 18:02:47	2	2006-02-16 02:30:53
6	16045	2005-08-23 22:25:26	772	14	2005-08-25 23:54:26	1	2006-02-16 02:30:53
7	16044	2005-08-23 22:24:39	1312	468	2005-08-25 04:08:39	1	2006-02-16 02:30:53
8	16043	2005-08-23 22:21:03	3869	526	2005-08-31 03:09:03	2	2006-02-16 02:30:53
9	16042	2005-08-23 22:20:40	629	131	2005-08-24 17:54:40	1	2006-02-16 02:30:53
10	16041	2005-08-23 22:20:26	4116	121	2005-08-25 20:14:26	2	2006-02-16 02:30:53
11	16040	2005-08-23 22:19:33	3524	195	2005-09-02 02:19:33	2	2006-02-16 02:30:53
12	16039	2005-08-23 22:18:51	545	78	2005-08-31 19:55:51	2	2006-02-16 02:30:53
13	16038	2005-08-23 22:14:31	2612	177	2005-08-30 03:28:31	1	2006-02-16 02:30:53

Total rows: 1000 of 16045 Query complete 00:00:00.152 Ln 1, Col 33

pgAdmin 4

127.0.0.1:5433/browser/

pgAdmin

File Object Tools Help

Dashboard Properties SQL Statistics Dependencies Dependents Processes DVD Rental/postgres@Postgres

Browsers (1)

Postgres

Databases (3)

DVD Rental

Castes

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

Subscriptions

University

postgres

Login/Group Roles

Tablespaces

Query

Query History

```
1 SELECT * FROM customer WHERE customer_id = 1;
```

Data Output Messages Notifications

	customer_id	store_id	first_name	last_name	email	address_id	activebool	create_date	last_update	active	count_check
	[PK] integer	smallint	character varying (40)	character varying (45)	character varying (50)	smallint	boolean	date	timestamp without time zone	integer	smallint
1	1	1	Mary	Smith	mary.smith@sakilacustomer.org	5	true	2006-02-14	2022-12-24 16:44:08.662989	1	1

Total rows: 1 of 1 Query complete 00:00:00.111 Ln 1, Col 46



pgAdmin 4

127.0.0.1:5433/browser/

pgAdmin

FileObjectToolsHelp

DashboardPropertiesSQLStatisticsDependenciesDependentsProcessesDVD Rental/postgres@Postgres*

Browsers (1)

Postgres

Databases (3)

DVD Rental

Castes

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

Subscriptions

University

postgres

Login/Group Roles

Tablespaces

QueryQuery History

1INSERT INTO rental VALUES (16051, '2022-12-24 20:00:00', 2, 1, '2023-01-01 20:00:00', 1);

2SELECT * FROM rental ORDER BY rental_id DESC;

Data OutputMessagesNotifications

rental_id	rental_date	inventory_id	customer_id	return_date	staff_id	last_update	
[PK] integer	timestamp without time zone	integer	smallint	timestamp without time zone	smallint	timestamp without time zone	
1	16051	2022-12-24 20:00:00	2	1	2023-01-01 20:00:00	1	2022-12-24 16:45:13.247006
2	16050	2022-12-24 20:00:00	1	1	2023-01-01 20:00:00	1	2022-12-24 16:44:08.662989
3	16049	2005-08-23 22:50:12	2666	393	2005-08-30 01:01:12	2	2006-02-16 02:30:53
4	16048	2005-08-23 22:43:07	2019	103	2005-08-31 21:33:07	1	2006-02-16 02:30:53
5	16047	2005-08-23 22:42:48	2088	114	2005-08-25 02:48:48	2	2006-02-16 02:30:53
6	16046	2005-08-23 22:26:47	4364	74	2005-08-27 18:02:47	2	2006-02-16 02:30:53
7	16045	2005-08-23 22:25:26	772	14	2005-08-25 23:54:26	1	2006-02-16 02:30:53
8	16044	2005-08-23 22:24:39	1312	468	2005-08-25 04:08:39	1	2006-02-16 02:30:53
9	16043	2005-08-23 22:21:03	3869	526	2005-08-31 03:09:03	2	2006-02-16 02:30:53
10	16042	2005-08-23 22:20:40	629	131	2005-08-24 17:54:40	1	2006-02-16 02:30:53
11	16041	2005-08-23 22:20:26	4116	121	2005-08-25 20:14:26	2	2006-02-16 02:30:53
12	16040	2005-08-23 22:19:33	3524	195	2005-09-02 02:19:33	2	2006-02-16 02:30:53
13	16039	2005-08-23 22:18:51	545	78	2005-08-31 19:55:51	2	2006-02-16 02:30:53

Total rows: 1000 of 16046Query complete 00:00:00.165Ln 2, Col 46

pgAdmin 4

127.0.0.1:5433/browser/

pgAdmin

File Object Tools Help

Dashboard Properties SQL Statistics Dependencies Dependents Processes DVD Rental/postgres@Postgres*

Browsers (1)

Postgres

Databases (3)

DVD Rental

Castes

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

Subscriptions

University

postgres

Login/Group Roles

Tablespaces

Query

Query History

```
1 SELECT * FROM customer WHERE customer_id = 1;
```

Data Output Messages Notifications

customer_id	store_id	first_name	last_name	email	address_id	activebool	create_date	last_update	active	count_check
1	1	Mary	Smith	mary.smith@sakilacustomer.org	5	true	2006-02-14	2022-12-24 16:45:13.247006	1	2

Total rows: 1 of 1 Query complete 00:00:00.081 Ln 1, Col 46



pgAdmin 4

127.0.0.1:5433/browser/

pgAdmin

File Object Tools Help

Dashboard Properties SQL Statistics Dependencies Dependents Processes DVD Rental/postgres@Postgres*

Browsers (1)

Postgres

Databases (3)

DVD Rental

Castes

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

Subscriptions

University

postgres

Login/Group Roles

Tablespaces

Query

Query History

```
1 INSERT INTO rental VALUES (16052, '2022-12-24 20:00:00', 3, 1, '2023-01-01 20:00:00', 1);
2 SELECT * FROM rental ORDER BY rental_id DESC;
```

Data Output Messages Notifications

	rental_id	rental_date	inventory_id	customer_id	return_date	staff_id	last_update
	[PK] integer	timestamp without time zone	integer	smallint	timestamp without time zone	smallint	timestamp without time zone
1	16052	2022-12-24 20:00:00		3	1 2023-01-08 20:00:00	1	2022-12-24 16:45:52.610143
2	16051	2022-12-24 20:00:00		2	1 2023-01-01 20:00:00	1	2022-12-24 16:45:13.247006
3	16050	2022-12-24 20:00:00		1	1 2023-01-01 20:00:00	1	2022-12-24 16:44:08.662989
4	16049	2005-08-23 22:50:12	2666	393	2005-08-30 01:01:12	2	2006-02-16 02:30:53
5	16048	2005-08-23 22:43:07	2019	103	2005-08-31 21:33:07	1	2006-02-16 02:30:53
6	16047	2005-08-23 22:42:48	2088	114	2005-08-25 02:48:48	2	2006-02-16 02:30:53
7	16046	2005-08-23 22:26:47	4364	74	2005-08-27 18:02:47	2	2006-02-16 02:30:53
8	16045	2005-08-23 22:25:26	772	14	2005-08-25 23:54:26	1	2006-02-16 02:30:53
9	16044	2005-08-23 22:24:39	1312	468	2005-08-25 04:08:39	1	2006-02-16 02:30:53
10	16043	2005-08-23 22:21:03	3869	526	2005-08-31 03:09:03	2	2006-02-16 02:30:53
11	16042	2005-08-23 22:20:40	629	131	2005-08-24 17:54:40	1	2006-02-16 02:30:53
12	16041	2005-08-23 22:20:26	4116	121	2005-08-25 20:14:26	2	2006-02-16 02:30:53
13	16040	2005-08-23 22:19:33	3574	195	2005-09-02 07:19:33	2	2006-02-16 02:30:53

Total rows: 1000 of 16047 Query complete 00:00:00.138 Ln 2, Col 46

pgAdmin 4

127.0.0.1:5433/browser/

pgAdmin

File Object Tools Help

Dashboard Properties SQL Statistics Dependencies Dependents Processes DVD Rental/postgres@Postgres*

Browsers (1)

Postgres

Databases (3)

DVD Rental

Castes

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

Subscriptions

University

postgres

Login/Group Roles

Tablespaces

Query

Query History

```
1 SELECT * FROM customer WHERE customer_id = 1;
```

Data Output Messages Notifications

	customer_id	store_id	first_name	last_name	email	address_id	activebool	create_date	last_update	active	count_check
	[PK] integer	smallint	character varying (40)	character varying (45)	character varying (50)	smallint	boolean	date	timestamp without time zone	integer	smallint
1	1	1	Mary	Smith	mary.smith@sakilacustomer.org	5	true	2006-02-14	2022-12-24 16:45:52.610143	1	0

Total rows: 1 of 1 Query complete 00:00:00.120 Ln 1, Col 46



```
1 WITH film_rental_amount AS (  
2     SELECT inventory.film_id AS film_id,  
3           SUM(amount) AS amount  
4 FROM payment,  
5       rental,  
6       inventory  
7 WHERE rental.rental_id = payment.rental_id  
8       AND inventory.inventory_id = rental.inventory_id  
9 GROUP BY inventory.film_id  
10 )  
11 SELECT title,  
12        rating,  
13        amount,  
14        RANK() OVER(ORDER BY amount DESC) AS overall_rank,  
15        RANK() OVER(PARTITION BY film.rating ORDER BY amount DESC) AS rating_rank,  
16        (  
17            CASE  
18                WHEN NTILE(4) OVER(  
19                    ORDER BY amount DESC  
20                ) = 1 THEN 'Yes'  
21                ELSE 'No'  
22            END  
23        ) AS first_quartile  
24 FROM film,  
25     film_rental_amount  
26 WHERE film_rental_amount.film_id = film.film_id  
27 ORDER BY film.title;
```

	title	rating	amount	overall_rank	rating_rank	first_quartile
	character varying (255)	numeric	numeric	bigint	bigint	text
1	Academy Dinosaur	PG	33.79	718	147	No
2	Ada Goodfinge	G	52.59	558	99	No
3	Adaptation Holes	NC-17	34.89	699	137	No
4	Affair Prejudice	G	83.79	261	47	No
5	African Egg	G	47.89	557	99	No
6	Agent Truman	PG	111.81	118	24	Yes
7	Angeline Stern	PG-13	82.85	263	64	No
8	Aspen Paddock	R	98.85	239	46	Yes
9	Aldonna Devil	PG-13	71.88	354	80	No
10	Aladdin Calendar	NC-17	131.77	66	18	Yes
11	Alamo Videotape	G	30.78	761	132	No
12	Alaska Phantom	PG	37.77	676	135	No
13	Alvin Carter	NC-17	94.89	791	61	No

Total rows: 958 of 958 Query complete 00:00:00.131 Ln 23, Col 32



```
1 SELECT DISTINCT rating,
2     SUM(amount) OVER(
3         PARTITION BY rating
4         ORDER BY date_part('month', payment_date) RANGE BETWEEN 1 PRECEDING AND 1 PRECEDING
5     ) as previous_month,
6     SUM(amount) OVER(
7         PARTITION BY rating,
8         date_part('month', payment_date)
9     ) as this_month,
10    SUM(amount) OVER(
11        PARTITION BY rating
12        ORDER BY date_part('month', payment_date) RANGE BETWEEN 1 FOLLOWING AND 1 FOLLOWING
13    ) as next_month,
14    date_part('month', payment_date) AS month
15 FROM film,
16     inventory,
17     rental,
18     payment
19 where film.film_id = inventory.film_id
20     AND inventory.inventory_id = rental.inventory_id
21     AND rental.rental_id = payment.rental_id
22 ORDER BY month;
```

The screenshot shows the pgAdmin 4 interface. The top pane displays the SQL query, and the bottom pane shows the query results. The results are organized into columns: rating, previous_month, this_month, next_month, and month. The data is presented in a table format with 13 rows of results.

rating	previous_month	this_month	next_month	month
G		1422.60	3944.62	2
PG		1658.99	4757.52	2
PG-13		1856.58	5316.63	2
R		1745.78	4782.76	2
NC-17		1667.89	5085.03	2
G	1422.60	3944.62	5040.03	3
PG	1658.99	4757.52	5725.45	3
PG-13	1856.58	5316.63	6563.76	3
R	1745.78	4782.76	5461.78	3
NC-17	1667.89	5085.03	5768.44	3
G	3944.62	5040.03	104.63	4
PG	4757.52	5725.45	94.69	4
PG-13	5316.63	6563.76	118.59	4