

Nomor Kelompok : 10

Kelas : K01

NIM : 18222013

Nama : Aththariq Lisan Q. D. S.

NIM : <isi dengan NIM>

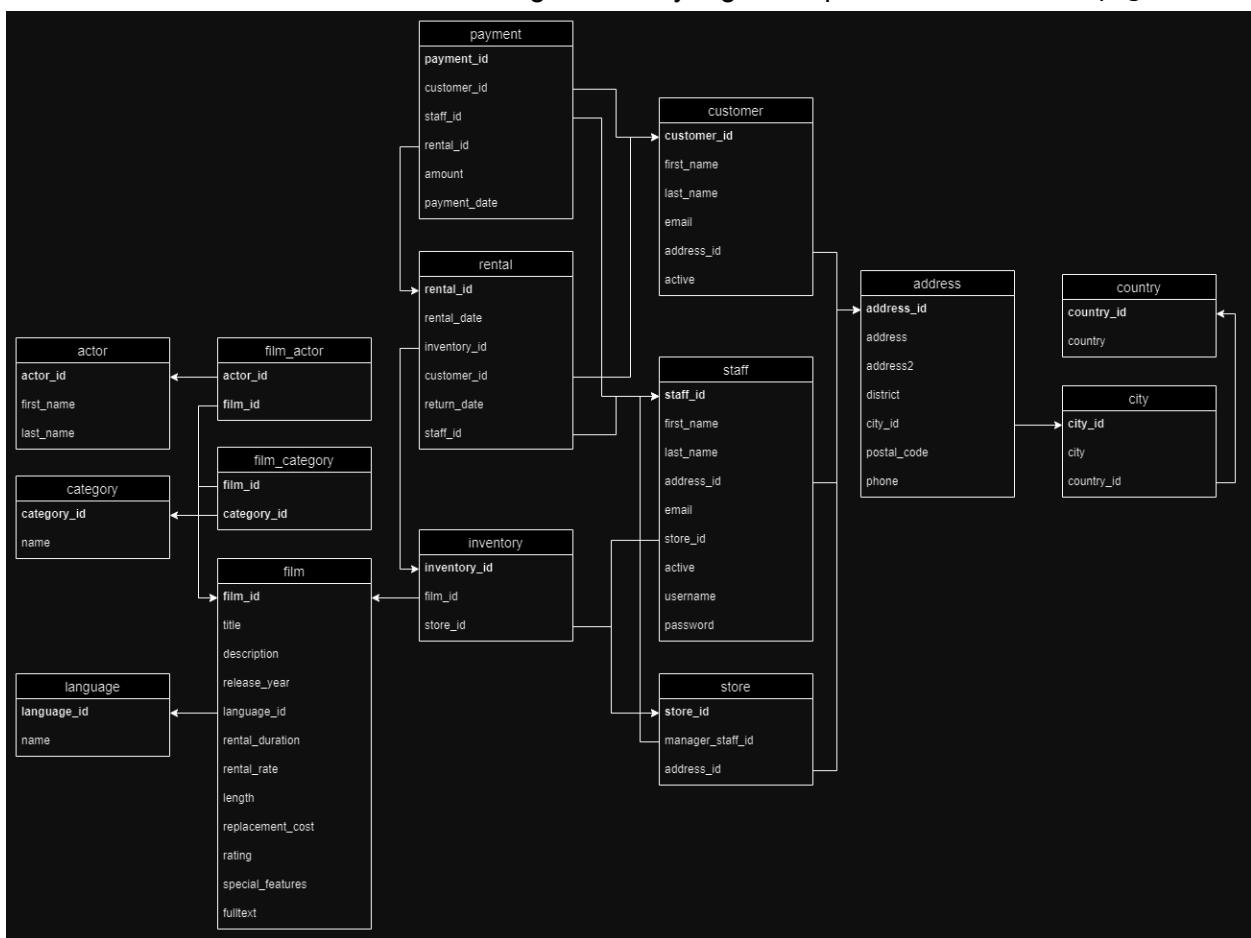
Nama : <isi dengan nama lengkap>

Lembar Kerja Praktikum 2 II2250 Manajemen Basis Data STI

Materi: *Schema Tuning & Index Tuning*

I. Skema Basis Data

Diberikan skema basis data sebagai berikut yang tersimpan dalam database pagila.



II. Soal

(Note: Pastikan telah terdapat database bernama pagila di dalam komputer yang digunakan. Jika belum, buatlah sebuah database bernama pagila dan import pagila.sql ke dalam database tersebut!)

1. Function

Toko pagila mengatur harga rental film dengan ketentuan 50% dari harga rental tersebut akan digunakan sebagai biaya operasional (sewa tempat, gaji pegawai, dsb), dan sisanya sebagai keuntungan rental tiap kali dilakukan penyewaan. Rental baru bisa dikatakan menghasilkan keuntungan ketika biaya beli film (80% dari biaya penggantian film) sudah tercover dari keuntungan yang didapatkan ketika pelanggan melakukan penyewaan atau dapat disebut sebagai break even point (BEP).

Buatlah sebuah function untuk menghitung break even point (berapa kali film disewa agar menutupi biaya pembelian film tersebut) dengan menerima argumen berupa harga beli dan keuntungan tiap melakukan rental film tersebut.

Lalu buatlah query untuk menampilkan judul film, tahun rilis, durasi rental, dan break even point (BEP) dari film tersebut dan urutkan berdasarkan BEP terkecil.

HINT : Bulatkan hasil fungsi BEP ke atas karena penyewaan hanya bisa dilakukan dalam bilangan bulat.

HINT TAMBAHAN : IF DALAM POSTGRESQL

Jawaban:

Query Pembuatan Function	CREATE FUNCTION calculate_bep(purchase_cost NUMERIC, profit_per_rental NUMERIC) RETURNS INTEGER AS \$\$ DECLARE bep INTEGER; BEGIN IF (0.8 * purchase_cost) <= (profit_per_rental * 1) THEN RETURN 1; ELSE bep := CEIL(purchase_cost / (0.5 * profit_per_rental)); RETURN bep; END IF; END; \$\$ LANGUAGE plpgsql;
--------------------------	--

SS Query Pembuatan Function

```
[pagila=# drop function calculate_bep;
DROP FUNCTION
pagila=# CREATE FUNCTION calculate_bep(purchase_cost NUMERIC, profit_per_rental NUMERIC)
RETURNS INTEGER AS $$
BEGIN
    DECLARE bep INTEGER;
    BEGIN
        IF (0.8 * purchase_cost) <= (profit_per_rental * 1) THEN
            RETURN 1;
        ELSE
            bep := CEIL(purchase_cost / (0.5 * profit_per_rental));
            RETURN bep;
        END IF;
    END;
$$ LANGUAGE plpgsql;
CREATE FUNCTION
pagila=# ]
```

Query Penerapan Function

```
SELECT
    film.title,
    film.release_year,
    film.rental_duration,
    calculate_bep(film.replacement_cost * 0.8,
    film.replacement_cost * 0.5) AS break_even_point
FROM
    film
ORDER BY
    break_even_point;
```

SS Hasil Penerapan Function

*boleh SS awal dan akhir row saja jika terlalu panjang

Wizard Coldblooded	2006	4		4
Wolves Desire	2006	7		4
Women Dorado	2006	4		4
Won Dares	2006	7		4
Wonderful Drop	2006	3		4
Wonderland Christmas	2006	4		4
Wonka Sea	2006	6		4
Words Hunter	2006	3		4
Worker Tarzan	2006	7		4
Working Microcosmos	2006	4		4
World Leathernecks	2006	3		4
Worst Banger	2006	4		4
Wrath Mile	2006	5		4
Wrong Behavior	2006	6		4
Wyoming Storm	2006	6		4
Yentl Idaho	2006	5		4
Young Language	2006	6		4
Youth Kick	2006	4		4
Zhivago Core	2006	6		4
Zoolander Fiction	2006	5		4
Zorro Ark	2006	3		4
(1000 rows)				

title	release_year	rental_duration	break_even_point
Chamber Italian	2006	7	4
Grosse Wonderful	2006	5	4
Airport Pollock	2006	6	4
Bright Encounters	2006	4	4
Academy Dinosaur	2006	6	4
Ace Goldfinger	2006	3	4
Adaptation Holes	2006	7	4
Affair Prejudice	2006	5	4
African Egg	2006	6	4
Agent Truman	2006	3	4
Airplane Sierra	2006	6	4
Alabama Devil	2006	3	4
Aladdin Calendar	2006	6	4
Alamo videotape	2006	6	4
Alaska Phantom	2006	6	4
Date Speed	2006	4	4
Ali Forever	2006	4	4
Alice Fantasia	2006	6	4
Alien Center	2006	5	4

2. Function

Toko Pagila berencana untuk menerapkan diskon pada biaya sewa film berdasarkan dua kriteria: **popularitas kategori film** dan **nilai rating** dari film tersebut. Diskon ditentukan dengan aturan berikut:

- Film yang termasuk dalam 5 besar kategori film terpopuler (kategori film yang paling banyak dipinjam) akan menerima diskon sebesar 20%.
- Film yang memiliki **rental rate** lebih dari 4 akan menerima diskon sebesar 5%.
- Film yang memenuhi kedua kriteria di atas akan menerima diskon sebesar 25%.

Tampilkan 10 film dengan diskon terbesar

Catatan : Diperbolehkan membuat view untuk membantu perhitungan

Jawaban:

Query Pembuatan Function

```
CREATE VIEW popular_category_counts AS
SELECT
    c.name AS category_name,
    COUNT(fc.film_id) AS film_count
FROM
    category c
JOIN film_category fc ON c.category_id =
fc.category_id
JOIN rental r ON fc.film_id = r.inventory_id
GROUP BY
```

```
c.name  
ORDER BY  
    film_count DESC;  
  
CREATE FUNCTION  
calculate_discount(category_name VARCHAR,  
rental_rate NUMERIC)  
RETURNS NUMERIC AS $$  
DECLARE  
    discount NUMERIC := 0;  
BEGIN  
    IF category_name IN (SELECT c.category_name  
FROM popular_category_counts c LIMIT 5) AND  
rental_rate > 4 THEN  
        RETURN 0.25;  
    ELSIF category_name IN (SELECT  
c.category_name FROM popular_category_counts c  
LIMIT 5) THEN  
        RETURN 0.2;  
    ELSIF rental_rate > 4 THEN  
        RETURN 0.05;  
    ELSE  
        RETURN 0;  
    END IF;  
END;  
$$ LANGUAGE plpgsql;
```

	<pre>Wrong Behavior 2006 6 4 Wyoming Storm 2006 6 4 Yentl Idaho 2006 5 4 Young Language 2006 6 4 Youth Kick 2006 4 4 Zhivago Core 2006 6 4 Zoolander Fiction 2006 5 4 Zorro Ark 2006 3 4 (1000 rows)</pre> <pre>pagila=# CREATE OR REPLACE VIEW popular_category_counts AS pagila=# SELECT pagila=# c.name AS category_name, pagila=# COUNT(fc.film_id) AS film_count pagila=# FROM pagila=# category c pagila=# JOIN film_category fc ON c.category_id = fc.category_id pagila=# JOIN rental r ON fc.film_id = r.inventory_id pagila=# GROUP BY pagila=# c.name pagila=# ORDER BY pagila=# film_count DESC; CREATE VIEW pagila=# </pre>
Query Penerapan Function	<pre>SELECT f.title, f.rental_rate, calculate_discount(c.name, f.rental_rate) AS discount FROM film f JOIN film_category fc ON f.film_id = fc.film_id JOIN category c ON fc.category_id = c.category_id ORDER BY discount DESC LIMIT 10;</pre>

SS Hasil Penerapan Function

```

pagila=# FROM
pagila=#   film f
pagila=# JOIN film_category fc ON f.film_id = fc.film_id
pagila=# JOIN category c ON fc.category_id = c.category_id
pagila=# ORDER BY
pagila=#   discount DESC
pagila=# LIMIT 10;

```

title	rental_rate	discount
Bright Encounters	4.99	0.25
Calendar Sunfight	4.99	0.25
Beauty Grease	4.99	0.25
Bilko Anonymous	4.99	0.25
Brooklyn Desert	4.99	0.25
Bubble Grosse	4.99	0.25
Apache Divine	4.99	0.25
Baby Hall	4.99	0.25
Aladdin Calendar	4.99	0.25
California Birds	4.99	0.25

(10 rows)

```

pagila=#
pagila=#

```

3. Function

Toko Pagila saat ini menjalin kerja sama dengan beberapa negara, diantaranya adalah India, Indonesia, dan Japan. Kerja sama ini akan berupaya untuk menyejahterakan staff Toko Pagila yang sering **melayani rental dari customer yang berasal dari negara-negara tersebut** dengan memberikan **insentif tambahan** dengan aturan sebagai berikut.

1. Setiap transaksi yang berasal dari negara **India**, maka staff akan diberikan insentif sebesar **1% dari amount transaksi**.
2. Setiap transaksi yang berasal dari negara **Indonesia**, maka staff akan diberikan insentif sebesar **0,5% dari amount transaksi**.
3. Setiap transaksi yang berasal dari negara **Japan**, maka staff akan diberikan insentif sebesar **2% dari amount transaksi**.

Buatlah sebuah function untuk menghitung **total insentif** yang didapatkan oleh setiap staff dengan menerima argumen berupa staff_id.

Lalu, buatlah query untuk menampilkan keseluruhan informasi dari tabel staff dan tampilkan juga total insentif yang mereka dapatkan.

Note: Dapat menggunakan CASE WHEN

Jawaban:

Query Pembuatan Function	<pre> CREATE OR REPLACE FUNCTION calculate_incentive(staff_id INT) RETURNS NUMERIC AS \$\$ DECLARE </pre>
---------------------------------	--

```

total_incentive NUMERIC := 0;
BEGIN
    total_incentive := (
        SELECT
            SUM(
                CASE
                    WHEN country.country = 'India' THEN
                        payment.amount * 0.01
                    WHEN country.country = 'Indonesia' THEN
                        payment.amount * 0.005
                    WHEN country.country = 'Japan' THEN
                        payment.amount * 0.02
                    ELSE 0
                END
            )
        FROM
            payment
        JOIN rental ON payment.rental_id =
            rental.rental_id
        JOIN customer ON rental.customer_id =
            customer.customer_id
        JOIN address ON customer.address_id =
            address.address_id
        JOIN city ON address.city_id = city.city_id
        JOIN country ON city.country_id =
            country.country_id
        WHERE
            payment.staff_id = calculate_incentive.staff_id
    );
    RETURN total_incentive;
END;
$$ LANGUAGE plpgsql;

```

```

aththariqilisan@aththariqilisan: ~ $ psql -c "\i runpsql.sh"

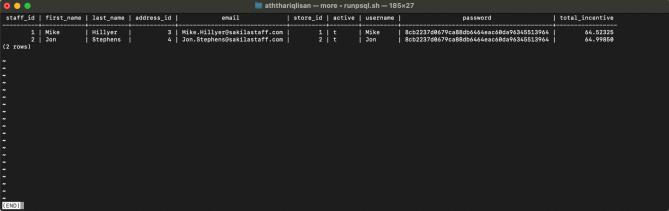
```

```

CREATE FUNCTION calculate_incentive()
RETURNS NUMERIC AS $$
BEGIN
    total_incentive := (
        SELECT
            SUM(
                CASE
                    WHEN country.country = 'India' THEN payment.amount * 0.01
                    WHEN country.country = 'Indonesia' THEN payment.amount * 0.005
                    WHEN country.country = 'Japan' THEN payment.amount * 0.02
                    ELSE 0
                END
            )
        FROM
            payment
        JOIN rental ON payment.rental_id = rental.rental_id
        JOIN customer ON rental.customer_id = customer.customer_id
        JOIN address ON customer.address_id = address.address_id
        JOIN city ON address.city_id = city.city_id
        JOIN country ON city.country_id = country.country_id
        WHERE
            payment.staff_id = calculate_incentive.staff_id
    );
    RETURN total_incentive;
END;
$$ LANGUAGE plpgsql;

```

Query Penerapan **SELECT**

Function	<pre>staff.*, calculate_incentive(staff.staff_id) AS total_incentive FROM staff;</pre>
SS Hasil Penerapan Function	 <pre>staff_id first_name last_name address_id email store_id active username password total_incentive 1 Mike Hiller 2 Mike.Hiller@nullstaff.com 1 t Mike 8c32376079ca80db044e4e0d86a963a5512964 66.52205 2 Don Stephens 2 Don.Stephens@nullstaff.com 1 t Don 8c323709097ca80db044e4e0d86a963a5512964 64.99900 (2 rows)</pre>

4. Procedure

Toko Pagila ingin memperbaiki sistem penyewaan film mereka dengan menambahkan fitur untuk mengelola transaksi penyewaan film secara otomatis. Oleh karena itu, Toko Pagila ingin membuat prosedur untuk menangani transaksi penyewaan film, termasuk pembuatan catatan penyewaan dan pembayaran.

- Buatlah prosedur untuk menangani transaksi penyewaan film. Prosedur ini harus menerima parameter seperti tanggal sewa, id inventory, id customer, id staff, durasi sewa, dan tarif sewa (rental rate). Prosedur ini harus secara otomatis menambahkan catatan penyewaan dan membuat catatan pembayaran apabila pembayaran dilakukan di depan (
- Jumlah yang dibayar didapatkan dari rental_duration * rental_rate
- Lakukan panggilan pada prosedur tersebut dengan parameter yang sesuai. Gunakan tanggal hari ini sebagai rental date.
- Buat query untuk menampilkan data dari tabel rental dan payment setelah prosedur berhasil dipanggil.

Jawaban:

Query Pembuatan Prosedur	<pre>CREATE PROCEDURE process_rental(rental_date DATE, inventory_id INT, customer_id INT, staff_id INT, rental_rate NUMERIC) LANGUAGE plpgsql AS \$\$ DECLARE rental_id INT; payment_id INT;</pre>
---------------------------------	--

```

total_amount NUMERIC;
duration INT;
RETURN_DATE DATE;
payment_date DATE;
BEGIN
    SELECT rental_duration INTO duration
    FROM film
    WHERE film_id = inventory_id;

    RETURN_DATE := rental_date + interval '1 day' *
duration;

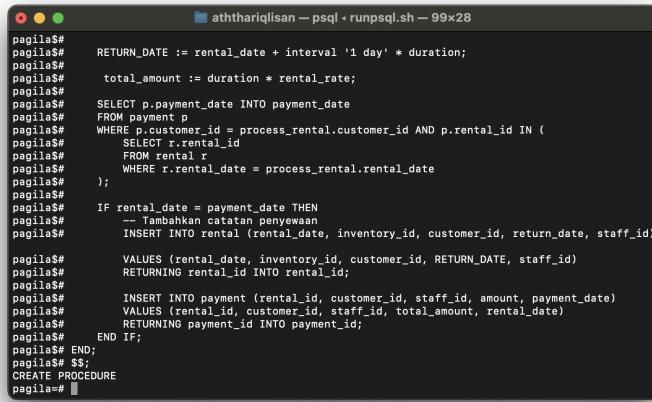
    total_amount := duration * rental_rate;

    SELECT p.payment_date INTO payment_date
    FROM payment p
    WHERE p.customer_id = process_rental.customer_id
AND p.rental_id IN (
    SELECT r.rental_id
    FROM rental r
    WHERE r.rental_date = process_rental.rental_date
);

    IF rental_date = payment_date THEN
        INSERT INTO rental (rental_date, inventory_id,
customer_id, return_date, staff_id)
        VALUES (rental_date, inventory_id, customer_id,
RETURN_DATE, staff_id)
        RETURNING rental_id INTO rental_id;

        INSERT INTO payment (rental_id, customer_id,
staff_id, amount, payment_date)
        VALUES (rental_id, customer_id, staff_id,
total_amount, rental_date)
        RETURNING payment_id INTO payment_id;
    END IF;
END;
$$;

```

SS Query Pembuatan Prosedur	 <pre> aththariqisan - psql < runpsql.sh - 99x28 pagila\$# RETURN_DATE := rental_date + interval '1 day' * duration; pagila\$# total_amount := duration * rental_rate; pagila\$# pagila\$# SELECT p.payment_date INTO payment_date pagila\$# FROM payment p pagila\$# WHERE p.customer_id = process_rental.customer_id AND p.rental_id IN (pagila\$# SELECT r.rental_id pagila\$# FROM rental r pagila\$# WHERE r.rental_date = process_rental.rental_date); pagila\$# pagila\$# IF rental_date = payment_date THEN -- Tambahkan catatan penyewaan INSERT INTO rental (rental_date, inventory_id, customer_id, return_date, staff_id) VALUES (rental_date, inventory_id, customer_id, RETURN_DATE, staff_id) RETURNING rental_id INTO rental_id; pagila\$# INSERT INTO payment (rental_id, customer_id, staff_id, amount, payment_date) VALUES (rental_id, customer_id, staff_id, total_amount, rental_date) RETURNING payment_id INTO payment_id; pagila\$# END IF; pagila\$# END; pagila\$# \$\$; CREATE PROCEDURE pagila\$# </pre>
Query Pemanggilan Prosedur	CALL process_rental(current_date, 1, 1, 1, 4.99); <pre> [pagila=# CALL process_rental(current_date, 1, 1, 1, 4.99); CALL pagila=# </pre>
Query untuk Menampilkan Data & SS	SELECT * FROM rental r JOIN payment p ON r.rental_id = p.rental_id ORDER BY rental_date desc LIMIT 5;

5. Procedure

Berkurangnya attention span masyarakat masa kini membuat manajemen Toko Pagila berpikir untuk meningkatkan promosi film dengan durasi pendek. Oleh karena itu, Toko Pagila ingin menandai film-film mana yang memiliki durasi kurang dari 60 menit sebagai film pendek untuk menjadi bahan marketing.

- Tambahkan atribut `is_short_movie` (boolean) pada tabel `film`. Dengan definisi durasi film pendek yang ditentukan manajemen, buatlah prosedur untuk memberikan value `is_short_movie` sesuai dengan apakah film tersebut termasuk film pendek (`true`) atau bukan (`false`).
- Lakukan panggilan pada prosedur tersebut!
- Buat query untuk menampilkan data dari tabel `film` setelah prosedur berhasil dipanggil.

Jawaban:

Query	ALTER TABLE film
--------------	------------------

Penambahan kolom	<pre>ADD COLUMN is_short_movie BOOLEAN; pagila=# ALTER TABLE film pagila=# ADD COLUMN is_short_movie BOOLEAN; ALTER TABLE</pre>
Query Pembuatan Prosedur	<pre>CREATE PROCEDURE short_movies() LANGUAGE plpgsql AS \$\$ BEGIN UPDATE film SET is_short_movie = CASE WHEN length < 60 THEN true ELSE false END; END; \$\$;</pre>
SS Query Pembuatan Prosedur	<pre>pagila=# CREATE PROCEDURE short_movies() pagila=# LANGUAGE plpgsql pagila=# AS \$\$ pagila#\$# BEGIN pagila#\$# UPDATE film pagila#\$# SET is_short_movie = CASE pagila#\$# WHEN length < 60 THEN true pagila#\$# ELSE false pagila#\$# END; pagila#\$# END; pagila#\$# \$\$; CREATE PROCEDURE</pre>
Query Pemanggilan Prosedur	<pre>CALL short_movies(); [pagila=# CALL short_movies(); CALL</pre>
Query untuk Menampilkan Data & SS	<pre>SELECT film_id, title, length, rental_rate, rating, is_short_movie FROM film;</pre>

6. Procedure

Toko Pagila ingin membuat strategi bisnis baru berupa membership untuk meningkatkan loyalitas customer mereka. Terdapat 2 kategori membership, yaitu premium dan standard. Membership premium memiliki benefit yang lebih banyak dibandingkan dengan membership standard. Untuk memudahkan pemberian benefit, toko pagila ingin melakukan penambahan atribut membership pada tabel customer. Customer yang termasuk dalam membership premium merupakan customer yang memiliki total amount payment di atas 160.00.

- a. Tambahkan atribut membership_category (varchar (25)) pada tabel customer. Buatlah prosedur untuk memberikan value membership_category sesuai dengan kategori membership (premium / standard) dari customer terkait.
- b. Lakukan panggilan pada prosedur tersebut.
- c. Buat query untuk menampilkan data dari tabel customer setelah prosedur berhasil dipanggil. Tampilkan juga data customer dengan kategori membership premium.

Jawaban:

Query Penambahan kolom	ALTER TABLE customer ADD COLUMN membership_category VARCHAR(25); <pre>pagila=# ALTER TABLE customer pagila-# ADD COLUMN membership_category VARCHAR(25); ALTER TABLE</pre>
Query Pembuatan Prosedur	CREATE PROCEDURE assign_membership_category() LANGUAGE plpgsql AS \$\$ BEGIN UPDATE customer c SET membership_category = 'premium' FROM (SELECT customer_id FROM payment GROUP BY customer_id HAVING SUM(amount) > 160.00) AS premium_customers WHERE c.customer_id = premium_customers.customer_id; UPDATE customer SET membership_category = 'standard' WHERE membership_category IS NULL;

	<pre>END; \$\$;</pre>																																																																																																																																					
SS Query Pembuatan Prosedur	<pre>pagila=# CREATE PROCEDURE assign_membership_category() pagila=# LANGUAGE plpgsql pagila=# AS \$\$ pagila## BEGIN pagila## UPDATE customer c pagila## SET membership_category = 'premium' pagila## FROM (pagila## SELECT customer_id pagila## FROM payment pagila## GROUP BY customer_id pagila## HAVING SUM(amount) > 160.00 pagila##) AS premium_customers pagila## WHERE c.customer_id = premium_customers.customer_id; pagila## UPDATE customer pagila## SET membership_category = 'standard' pagila## WHERE membership_category IS NULL; pagila## END; pagila## \$\$; CREATE PROCEDURE "";</pre>																																																																																																																																					
Query Pemanggilan Prosedur	<p>CALL assign_membership_category();</p> <pre>pagila=# CALL assign_membership_category(); CALL</pre>																																																																																																																																					
Query untuk Menampilkan Data & SS	<p>SELECT * FROM customer;</p> <table border="1"> <thead> <tr> <th>customer_id</th> <th>first_name</th> <th>last_name</th> <th>email</th> <th>address_id</th> <th>active</th> <th>membership_category</th> </tr> </thead> <tbody> <tr><td>137</td><td>Rhonda</td><td>Kennedy</td><td>rhonda.kennedy@sakilacustomer.org</td><td>141</td><td>t</td><td>premium</td></tr> <tr><td>144</td><td>Clara</td><td>Shaw</td><td>clara.shaw@sakilacustomer.org</td><td>151</td><td>t</td><td>premium</td></tr> <tr><td>145</td><td>Albert</td><td>Sant</td><td>albert.sant@sakilacustomer.org</td><td>152</td><td>t</td><td>premium</td></tr> <tr><td>178</td><td>Marion</td><td>Snyder</td><td>marion.snyder@sakilacustomer.org</td><td>182</td><td>t</td><td>premium</td></tr> <tr><td>181</td><td>Ana</td><td>Bradley</td><td>ana.bradley@sakilacustomer.org</td><td>185</td><td>t</td><td>premium</td></tr> <tr><td>223</td><td>Marcia</td><td>Dean</td><td>marcia.dean@sakilacustomer.org</td><td>228</td><td>t</td><td>standard</td></tr> <tr><td>403</td><td>Mike</td><td>Wise</td><td>mike.wise@sakilacustomer.org</td><td>408</td><td>t</td><td>premium</td></tr> <tr><td>410</td><td>Curtis</td><td>Irby</td><td>curtis.irby@sakilacustomer.org</td><td>415</td><td>t</td><td>premium</td></tr> <tr><td>449</td><td>Jenny</td><td>Alvarez</td><td>jenny.alvarez@sakilacustomer.org</td><td>464</td><td>t</td><td>premium</td></tr> <tr><td>522</td><td>Arnold</td><td>Havens</td><td>arnold.havens@sakilacustomer.org</td><td>528</td><td>t</td><td>premium</td></tr> <tr><td>526</td><td>Karl</td><td>Seal</td><td>karl.seal@sakilacustomer.org</td><td>532</td><td>t</td><td>premium</td></tr> <tr><td>624</td><td>Jared</td><td>Ely</td><td>jared.ely@sakilacustomer.org</td><td>630</td><td>t</td><td>standard</td></tr> <tr><td>1</td><td>Leoth</td><td>Smith</td><td>leoth.smith@sakilacustomer.org</td><td>8</td><td>t</td><td>standard</td></tr> <tr><td>2</td><td>Patricia</td><td>Johnson</td><td>patricia.johnson@sakilacustomer.org</td><td>6</td><td>t</td><td>standard</td></tr> <tr><td>3</td><td>Linda</td><td>Williams</td><td>linda.williams@sakilacustomer.org</td><td>7</td><td>t</td><td>standard</td></tr> <tr><td>4</td><td>Stephanie</td><td>Brown</td><td>stephanie.brown@sakilacustomer.org</td><td>7</td><td>t</td><td>standard</td></tr> <tr><td>5</td><td>Elizabeth</td><td>Brown</td><td>elizabeth.brown@sakilacustomer.org</td><td>9</td><td>t</td><td>standard</td></tr> <tr><td>6</td><td>Jennifer</td><td>Davis</td><td>jennifer.davis@sakilacustomer.org</td><td>10</td><td>t</td><td>standard</td></tr> </tbody> </table>	customer_id	first_name	last_name	email	address_id	active	membership_category	137	Rhonda	Kennedy	rhonda.kennedy@sakilacustomer.org	141	t	premium	144	Clara	Shaw	clara.shaw@sakilacustomer.org	151	t	premium	145	Albert	Sant	albert.sant@sakilacustomer.org	152	t	premium	178	Marion	Snyder	marion.snyder@sakilacustomer.org	182	t	premium	181	Ana	Bradley	ana.bradley@sakilacustomer.org	185	t	premium	223	Marcia	Dean	marcia.dean@sakilacustomer.org	228	t	standard	403	Mike	Wise	mike.wise@sakilacustomer.org	408	t	premium	410	Curtis	Irby	curtis.irby@sakilacustomer.org	415	t	premium	449	Jenny	Alvarez	jenny.alvarez@sakilacustomer.org	464	t	premium	522	Arnold	Havens	arnold.havens@sakilacustomer.org	528	t	premium	526	Karl	Seal	karl.seal@sakilacustomer.org	532	t	premium	624	Jared	Ely	jared.ely@sakilacustomer.org	630	t	standard	1	Leoth	Smith	leoth.smith@sakilacustomer.org	8	t	standard	2	Patricia	Johnson	patricia.johnson@sakilacustomer.org	6	t	standard	3	Linda	Williams	linda.williams@sakilacustomer.org	7	t	standard	4	Stephanie	Brown	stephanie.brown@sakilacustomer.org	7	t	standard	5	Elizabeth	Brown	elizabeth.brown@sakilacustomer.org	9	t	standard	6	Jennifer	Davis	jennifer.davis@sakilacustomer.org	10	t	standard
customer_id	first_name	last_name	email	address_id	active	membership_category																																																																																																																																
137	Rhonda	Kennedy	rhonda.kennedy@sakilacustomer.org	141	t	premium																																																																																																																																
144	Clara	Shaw	clara.shaw@sakilacustomer.org	151	t	premium																																																																																																																																
145	Albert	Sant	albert.sant@sakilacustomer.org	152	t	premium																																																																																																																																
178	Marion	Snyder	marion.snyder@sakilacustomer.org	182	t	premium																																																																																																																																
181	Ana	Bradley	ana.bradley@sakilacustomer.org	185	t	premium																																																																																																																																
223	Marcia	Dean	marcia.dean@sakilacustomer.org	228	t	standard																																																																																																																																
403	Mike	Wise	mike.wise@sakilacustomer.org	408	t	premium																																																																																																																																
410	Curtis	Irby	curtis.irby@sakilacustomer.org	415	t	premium																																																																																																																																
449	Jenny	Alvarez	jenny.alvarez@sakilacustomer.org	464	t	premium																																																																																																																																
522	Arnold	Havens	arnold.havens@sakilacustomer.org	528	t	premium																																																																																																																																
526	Karl	Seal	karl.seal@sakilacustomer.org	532	t	premium																																																																																																																																
624	Jared	Ely	jared.ely@sakilacustomer.org	630	t	standard																																																																																																																																
1	Leoth	Smith	leoth.smith@sakilacustomer.org	8	t	standard																																																																																																																																
2	Patricia	Johnson	patricia.johnson@sakilacustomer.org	6	t	standard																																																																																																																																
3	Linda	Williams	linda.williams@sakilacustomer.org	7	t	standard																																																																																																																																
4	Stephanie	Brown	stephanie.brown@sakilacustomer.org	7	t	standard																																																																																																																																
5	Elizabeth	Brown	elizabeth.brown@sakilacustomer.org	9	t	standard																																																																																																																																
6	Jennifer	Davis	jennifer.davis@sakilacustomer.org	10	t	standard																																																																																																																																

III. Pembagian Tugas

NIM	Nama	Tugas
18222013	Aththariq Lisan Q. D. S.	