Nama:Faisal Rizky NPM:1406573015
Kelas : C Asdos :F

I. Stored Procedure

Menambahkan atribut frekuensi_dipinjam pada relasi KOMIK dengan menggunakan perintah

ALTER TABLE KOMIK ADD COLUMN frekuensi_dipinjam INT DEFAULT 0;

```
faisal.rizky=> ALTER TABLE KOMIK ADD COLUMN frekuensi_dipinjam INT DEFAULT
faisal.rizky-> 0;
ALTER TABLE
```

Membuat stored procedure untuk menghitung frekuensi suatu komik dipinjam pada relasi KOMIK dalam skema SIRENKOM.

CREATE OR REPLACE FUNCTION hitung_frekuensi_dipinjam (no_kmk CHAR(6))RETURNS INTEGER AS

$$ DECLARE frekuensi INTEGER; BEGIN SELECT COUNT(DP.*) INTO frekuensi

FROM KOMIK K, DETAIL_PEMINJAMAN DP

WHERE K.no_komik = no_kmk AND DP.no_komik = no_kmk;

UPDATE KOMIK K SET frekuensi_dipinjam = frekuensi

WHERE K.no_komik = no_kmk;

RETURN frekuensi;

END;

$$ LANGUAGE plpgsql;

```
faisal.rizky=> CREATE OR REPLACE FUNCTION hitung_frekuensi_dipinjam
faisal.rizky-> (no_kmk CHAR(6))RETURNS INTEGER AS
faisal.rizky-> $$
faisal.rizky$>  DECLARE
faisal.rizky$>  frekuensi INTEGER;
faisal.rizky$>  BEGIN
faisal.rizky$>  SELECT COUNT(DP.*) INTO frekuensi
faisal.rizky$>  FROM KOMIK K, DETAIL_PEMINJAMAN DP
faisal.rizky$>  WHERE K.no_komik = no_kmk AND DP.no_komik = no_kmk;
faisal.rizky$>  UPDATE KOMIK K SET frekuensi_dipinjam = frekuensi
faisal.rizky$>  WHERE K.no_komik = no_kmk;
faisal.rizky$>  RETURN frekuensi;
faisal.rizky$>  END;
faisal.rizky$> $$
faisal.rizky-> LANGUAGE plpgsql;
CREATE FUNCTION
```

Melihat stored procedure dengan menggunakan perintah (\df)

```
                                  List of functions
  Schema  |           Name            | Result data type | Argument data types |  Type
----------+---------------------------+------------------+---------------------+--------
 sirenkom | hitung_frekuensi_dipinjam | integer          | no_kmk character    | normal
(1 row)
```

Nama:Faisal Rizky NPM:1406573015
Kelas : C Asdos :F

Menjalankan stored procedure yang sudah dibuat, dapat dilakukan dengan perintah
SELECT hitung_frekuensi_dipinjam('K00003');

```
faisal.rizky=> SELECT hitung_frekuensi_dipinjam('K00003');
 hitung_frekuensi_dipinjam
---------------------------
                        71
(1 row)
```

Menghapus procedure hitung_frekuensi_dipinjam, menggunakan perintah
DROP FUNCTION hitung_frekuensi_dipinjam(CHAR(6));

```
faisal.rizky=> DROP FUNCTION hitung_frekuensi_dipinjam(CHAR(6));
DROP FUNCTION
```

Menghitung semua frekuensi dipinjam dari seluruh komik yang ada, dengan menggunakan looping pada stored procedure.

```
CREATE OR REPLACE FUNCTION hitung_semua_frekuensi() RETURNS void AS
$$
DECLARE row RECORD;
BEGIN
FOR row IN
SELECT K.no_komik as no_kmk, COUNT(DP.*) AS frekuensi
FROM KOMIK K, DETAIL_PEMINJAMAN DP
WHERE K.no_komik = DP.no_komik
GROUP BY K.no_komik
LOOP
UPDATE KOMIK SET frekuensi_dipinjam = row.frekuensi
 WHERE no_komik = row.no_kmk;
 END LOOP;
 END;
$$
 LANGUAGE plpgsql;
```

```
faisal.rizky=> CREATE OR REPLACE FUNCTION hitung_semua_frekuensi()
faisal.rizky-> RETURNS void AS
faisal.rizky-> $$
faisal.rizky$>  DECLARE
faisal.rizky$>  row RECORD;
faisal.rizky$>  BEGIN
faisal.rizky$>  FOR row IN
faisal.rizky$>  SELECT K.no_komik as no_kmk, COUNT(DP.*) AS
faisal.rizky$> frekuensi
faisal.rizky$>  FROM KOMIK K, DETAIL_PEMINJAMAN DP
faisal.rizky$>  WHERE K.no_komik = DP.no_komik
faisal.rizky$>  GROUP BY K.no_komik
faisal.rizky$>  LOOP
faisal.rizky$>  UPDATE KOMIK SET frekuensi_dipinjam =
faisal.rizky$> row.frekuensi
faisal.rizky$>  WHERE no_komik = row.no_kmk;
faisal.rizky$>  END LOOP;
faisal.rizky$>  END;
faisal.rizky$> $$
faisal.rizky-> LANGUAGE plpgsql;
CREATE FUNCTION
```

Nama:Faisal Rizky NPM:1406573015
Kelas : C Asdos :F

Mengeksekusi fungsi hitung_semua_frekuensi dengan perintah
SELECT hitung_semua_frekuensi();

```
faisal.rizky=> SELECT hitung_semua_frekuensi();
 hitung_semua_frekuensi
------------------------

(1 row)
```

Memeriksa data frekuensi apakah sudah diperbarui dengan perintah
SELECT frekuensi_dipinjam FROM KOMIK;

```
faisal.rizky=> SELECT frekuensi_dipinjam FROM KOMIK;
 frekuensi_dipinjam
--------------------
                 45
                  5
                  5
                 43
                 38
                  7
                 49
                  5
                 77
                 21
                 11
                  8
                 40
                  4
                 71
                 25
                 52
                  8
                  9
                 84
                 10
                 32
                 26
                 73
                  2
                 10
                  4
                 52
                  6
                 14
                 23
                 10
                 32
                  5
                 19
                 10
                 91
                 46
                 14
                 77
(40 rows)
```

II. Trigger

Membuat trigger setiap ada kejadian INSERT terhadap tabel DETAIL_PEMINJAMAN untuk mencegah input untuk atribut no_volume lebih besar dari jumlah_volume komik yang dipinjam.

Nama:Faisal Rizky NPM:1406573015
Kelas : C Asdos :F

```
CREATE OR REPLACE FUNCTION volume_constraint() RETURNS trigger AS
$$ DECLARE Jml_volume INTEGER;
 BEGIN
SELECT jumlah_volume INTO jml_volume
FROM KOMIK K
WHERE K.no_komik = NEW.no_komik;
 IF (TG_OP = 'INSERT') THEN IF(NEW.no_volume > jml_volume)
THEN RAISE EXCEPTION 'Error : Nomor volume komik tidak ada';
END IF;
RETURN NEW;
 END IF;
 END;
 $$
LANGUAGE plpgsql;
```

```
 aisal.rizky=> CREATE OR REPLACE FUNCTION volume_constraint() RETURNS trigger AS
faisal.rizky-> $$ DECLARE Jml_volume INTEGER;
faisal.rizky$>  BEGIN
faisal.rizky$> SELECT jumlah_volume INTO jml_volume
faisal.rizky$> FROM KOMIK K
faisal.rizky$> WHERE K.no_komik = NEW.no_komik;
faisal.rizky$>  IF (TG_OP = 'INSERT') THEN IF(NEW.no_volume > jml_volume)
faisal.rizky$> THEN RAISE EXCEPTION 'Error : Nomor volume komik tidak ada';
faisal.rizky$> END IF;
faisal.rizky$> RETURN NEW;
faisal.rizky$>  END IF;
faisal.rizky$>  END;
faisal.rizky$>  $$
faisal.rizky-> LANGUAGE plpgsql;
CREATE FUNCTION
```

Membuat trigger yang memanggil stored procedure “volume_constraint” dengan perintah

```
CREATE TRIGGER volume_trigger
AFTER INSERT ON DETAIL_PEMINJAMAN FOR EACH ROW
EXECUTE PROCEDURE volume_constraint();
```

```
faisal.rizky=> CREATE TRIGGER volume_trigger
faisal.rizky-> AFTER INSERT
faisal.rizky-> ON DETAIL_PEMINJAMAN FOR EACH ROW
faisal.rizky-> EXECUTE PROCEDURE volume_constraint();
CREATE TRIGGER
```

Memeriksa apakah trigger berjalan dengan benar

```
INSERT INTO DETAIL_PEMINJAMAN VALUES ('388', ‘K00001’, 50, NULL, 0);
```

```
faisal.rizky=> INSERT INTO DETAIL_PEMINJAMAN VALUES ('388','K00001', 50,
NULL, 0);
ERROR:  Error : Nomor volume komik tidak ada
```

Nama:Faisal Rizky NPM:1406573015
Kelas : C Asdos :F

III. Latihan

1. ALTER TABLE ANGGOTA ADD COLUMN status BOOLEAN DEFAULT TRUE;

```
faisal.rizky=> ALTER TABLE ANGGOTA ADD COLUMN status BOOLEAN DEFAULT  TRUE;
ALTER TABLE
```

```
CREATE OR REPLACE FUNCTION status()
RETURNS void AS $$
DECLARE row RECORD;
BEGIN
FOR row IN
SELECT K.no_komik as no_kmk, COUNT(DP.*) AS frekuensi
FROM KOMIK K, DETAIL_PEMINJAMAN DP
WHERE K.no_komik = DP.no_komik
GROUP BY K.no_komik LOOP UPDATE KOMIK SET frekuensi_dipinjam = row.frekuensi
WHERE no_komik = row.no_kmk; END LOOP; END; $$ LANGUAGE plpgsql;
```

2. Menampikan judul dan jumlah frekuensi dipinjam komik berjudul Doraemon dan Detective Conan dengan menggunakan perintah

```
SELECT judul, frekuensi_dipinjam
FROM KOMIK
WHERE judul = ‘Doraemon’ OR judul=’Detective Conan’;
```