-- C/D detailed table creation and raw data extraction

CREATE TABLE actor\_dollars\_detail (

actor\_id INT,

name VARCHAR(255),

amount NUMERIC(5,2),

movie\_title VARCHAR(255),

date VARCHAR(12)

);

INSERT INTO actor\_dollars\_detail

SELECT actor.actor\_id, Full\_name(first\_name, last\_name), amount, title, Month(rental\_date)

FROM payment

INNER JOIN rental

ON rental.customer\_id = payment.customer\_id

INNER JOIN inventory

ON inventory.inventory\_id = rental.inventory\_id

INNER JOIN film

ON film.film\_id = inventory.film\_id

INNER JOIN film\_actor

ON film\_actor.film\_id = film.film\_id

INNER JOIN actor

ON actor.actor\_id = film\_actor.actor\_id

WHERE rental\_date BETWEEN '2005-06-01 00:00:00' AND '2005-08-31 00:00:00';

DROP TABLE actor\_dollars\_detail;

SELECT \*

FROM actor\_dollars\_detail;

SELECT \*

FROM actor\_dollars\_detail

WHERE amount = 12.99;

-- C. summary table

INSERT INTO top\_ten

SELECT name, TO\_CHAR(SUM(amount), 'FM999,999,999') AS total\_sales

FROM actor\_dollars\_detail

GROUP BY name

ORDER BY SUM(amount) DESC

LIMIT 10;

--another option for analysts to use from the detailed table

SELECT movie\_title, COUNT(movie\_title), SUM(amount)

FROM actor\_dollars\_detail

WHERE date = 'Jun 14, 2005'

GROUP BY movie\_title

ORDER BY COUNT(movie\_title) DESC;

CREATE TABLE top\_ten (

Name VARCHAR(255),

Total\_sales TEXT

);

SELECT \*

FROM top\_ten;

DROP TABLE top\_ten;

--B. function for transformation of first and last name from part A4

CREATE OR REPLACE FUNCTION full\_name(first\_name VARCHAR(45), last\_name VARCHAR(45))

RETURNS VARCHAR(255)

LANGUAGE plpgsql

AS

$$

BEGIN

RETURN first\_name || ' ' || last\_name;

END;

$$;

--B. function for transformation of date to MDY from A4

CREATE OR REPLACE FUNCTION Month(rental\_date TIMESTAMP)

RETURNS VARCHAR(12)

LANGUAGE plpgsql

AS $$

DECLARE RentalDate VARCHAR(12);

BEGIN

SELECT TO\_CHAR(rental\_date, 'FMMon DD, YYYY') INTO RentalDate;

RETURN RentalDate;

END;

$$;

-- E. function for trigger for detailed table to update the summary table

CREATE OR REPLACE FUNCTION insert\_trigger\_function()

RETURNS TRIGGER

LANGUAGE plpgsql

AS

$$

BEGIN

DELETE FROM top\_ten;

INSERT INTO top\_ten

SELECT name, TO\_CHAR(SUM(amount), 'FM999,999,999') AS total\_sales

FROM actor\_dollars\_detail

GROUP BY name

ORDER BY SUM(amount) DESC

LIMIT 10;

RETURN NEW;

END;

$$;

--E. trigger - added delete as to not mess up the original tables

CREATE TRIGGER weekly\_inserts

AFTER INSERT OR DELETE

ON actor\_dollars\_detail

FOR EACH STATEMENT

EXECUTE PROCEDURE insert\_trigger\_function()

--E. used to show that the trigger function works on insert and on delete

INSERT INTO actor\_dollars\_detail VALUES (110, 'Susan Davis', 12.99, 'Wash Heavenly');

DELETE FROM actor\_dollars\_detail

WHERE amount = 12.99;

DROP TRIGGER weekly\_inserts ON actor\_dollars\_detail;

--F. drop tables and test to make sure tables do not exist

CALL refresh \_table()

DROP PROCEDURE refresh\_table()

--F. refresh both tables

CREATE OR REPLACE PROCEDURE refresh\_table()

LANGUAGE plpgsql

AS $$

BEGIN

DELETE FROM actor\_dollars\_detail;

DELETE FROM top\_ten;

INSERT INTO actor\_dollars\_detail

SELECT actor.actor\_id, Full\_name(first\_name, last\_name) AS name, amount, title, Month(rental\_date) AS date

FROM payment

INNER JOIN rental

ON rental.customer\_id = payment.customer\_id

INNER JOIN inventory

ON inventory.inventory\_id = rental.inventory\_id

INNER JOIN film

ON film.film\_id = inventory.film\_id

INNER JOIN film\_actor

ON film\_actor.film\_id = film.film\_id

INNER JOIN actor

ON actor.actor\_id = film\_actor.actor\_id

WHERE rental\_date BETWEEN '2005-06-01 00:00:00' AND '2005-08-31 00:00:00';

INSERT INTO top\_ten

SELECT name, TO\_CHAR(SUM(amount), 'FM999,999,999') AS total\_sales

FROM actor\_dollars\_detail

GROUP BY name

ORDER BY SUM(amount) DESC

LIMIT 10;

RETURN;

END;

$$;