

```
-----  
-- Example 1  
-- first function in PL/R  
-----
```

```
CREATE OR REPLACE FUNCTION rsquare(val int) RETURNS int AS  
$$  
    return(val * val) # this line is R code  
$$  
LANGUAGE 'plr';
```

```
-- now rsquare() can be used as:  
SELECT rsquare(5);
```

```
-- yielding:  
-- plr=# SELECT rsquare(5);  
-- rsquare  
-- -----  
--      25  
-- (1 row)
```

```
-----  
-- Example 2  
-- generate a data frame inside R and use it as if it was a SQL table  
-----
```

```
DROP TYPE IF EXISTS rcircle;  
CREATE TYPE rcircle as (  
    radius double precision,  
    area double precision  
);
```

```
CREATE OR REPLACE FUNCTION rcircle_area(maxrad int) RETURNS SETOF rcircle AS  
$$  
    # R can perform operations on vectors (no need for loops)  
    r <- seq(1, maxrad)  
    a <- r * r * pi  
    result <- data.frame(radius = r, area = a)  
    return(result)  
$$  
LANGUAGE 'plr';
```

```
-- do this to compute all areas for r=1, 2, ... 20:  
SELECT * FROM rcircle_area(20);
```

```
-- yielding:  
-- plr=# SELECT * FROM rcircle_area(20);  
-- radius | area  
-- -----+-----  
--      1 | 3.14159265358979  
--      2 | 12.5663706143592  
-- ...  
--     20 | 1256.63706143592  
-- (20 rows)
```

```
-----  
-- Example 3  
-- get data from an SQL table into a R data frame and do some computation  
-----
```

```
DROP TABLE IF EXISTS rmeasurement;  
CREATE TABLE rmeasurement (  
    --
```

```

id      INT,
current DOUBLE PRECISION,
voltage DOUBLE PRECISION
);

INSERT INTO rmeasurement VALUES (1, 1.0, 2.498),
                                (2, 1.5, 3.753),
                                (3, 2.0, 5.002),
                                (4, 2.5, 6.247);

CREATE OR REPLACE FUNCTION rcomp_resistance() RETURNS double precision AS
$$
  dat <- pg.spi.exec("SELECT voltage, current FROM rmeasurement")
  # Ohm's law: voltage = current * resistance
  fit <- lm(voltage ~ current, data = dat)
  c <- coef(fit)
  # the second coefficient is the slope (= resistance)
  return(c[2])
$$
LANGUAGE 'plr';

-- this will yield the resistance:
SELECT rcomp_resistance();

-- yielding:
-- plr=# SELECT rcomp_resistance();
-- rcomp_resistance
-- -----
--          2.4992
-- (1 row)

-----
-- Example 4
-- generate a server-side plot
-- (not all graph packages work without X11, bitmap() does: it uses ghostscript)
-----

CREATE OR REPLACE FUNCTION rplot() RETURNS INT AS
$$
  fname <- "/var/www/graph/trig.png"
  bitmap(file = fname, type = "png256", width = 640, height = 480, units = "px")
  xmin <- -pi
  xmax <- pi
  ymin <- -1.2
  ymax <- 1.2
  curve(sin(x), col = "red", xlim = c(xmin, xmax), ylim = c(ymin, ymax),      \\.
        n = 640, ylab = "y")
  curve(cos(x), col = "blue", xlim = c(xmin, xmax), ylim = c(ymin, ymax),    \\.
        add = TRUE, n = 640, ylab = "y")
  curve(tan(x), col = "gray", xlim = c(xmin, xmax), ylim = c(ymin, ymax),   \\.
        add = TRUE, n = 640, ylab = "y")
  dev.off()
  system(paste("chmod 640", fname), ignore.stderr = TRUE)
  return(0)
$$
language 'plr';

-- this will create trig.png in /var/www/graph/:
SELECT rplot();

```