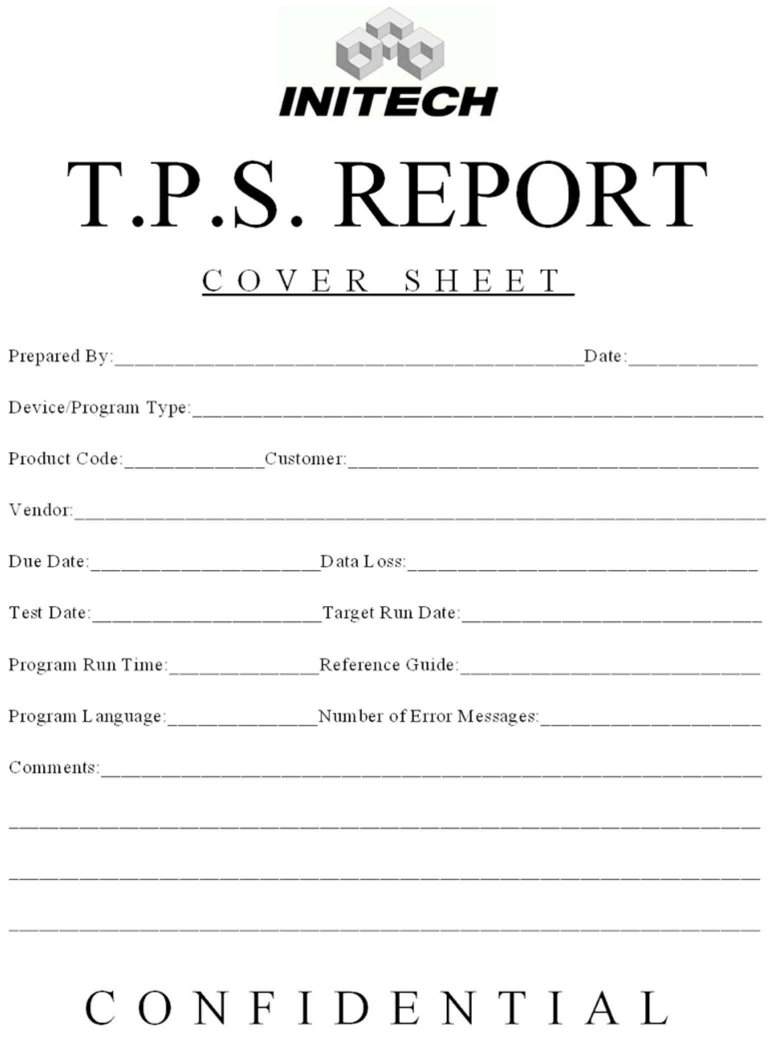
Windowing Functions

Thanks, Harada-San!

The Big Idea:

- Slices (Windows) of data
  - Handled independently
  - Can be tied back

Reporting



INITECH

T.P.S. REPORT

COVER SHEET

Prepared By:______________________________Date:__________

Device/Program Type:____

Product Code:__________Customer:______________________________

Vendor:____

Due Date:________________Data Loss:__________________________

Test Date:________________Target Run Date:_____________________

Program Run Time:__________Reference Guide:____________________

Program Language:__________Number of Error Messages:______________

Comments:____

____

____

____

CONFIDENTIAL

ROW_NUMBER (Before)

```
SELECT
    e1.empno,
    e1.depname,
    e1.salary,
    count(*) AS row_number
FROM
    empsalary e1
JOIN
    empsalary e2
    ON (e1.empno < e2.empno)
GROUP BY e1.empno, e1.depname, e1.salary
ORDER BY e1.empno DESC;
```

ROW_NUMBER (Before)

OOPS!

```
 empno |  depname  | salary | row_number
-------+-----------+--------+------------
     8 | develop   |   6000 |          1
     6 | sales     |   5500 |          2
    11 | develop   |   5200 |          4
    10 | develop   |   5200 |          4
     1 | sales     |   5000 |          5
     3 | sales     |   4800 |          7
     4 | sales     |   4800 |          7
     9 | develop   |   4500 |          8
     7 | develop   |   4200 |          9
     2 | personnel |   3900 |         10
     5 | personnel |   3500 |         11
(11 rows)
```

ROW_NUMBER (After)

```
SELECT
    empno,
    depname,
    salary,
    row_number() OVER (
        ORDER BY salary DESC NULLS LAST
    )
FROM
    empsalary
ORDER BY salary DESC;
```

ROW_NUMBER (After)

Yippee!

```
 empno |  depname  | salary | row_number
-------+-----------+--------+------------
     8 | develop   |   6000 |          1
     6 | sales     |   5500 |          2
    10 | develop   |   5200 |          3
    11 | develop   |   5200 |          4
     1 | sales     |   5000 |          5
     3 | sales     |   4800 |          6
     4 | sales     |   4800 |          7
     9 | develop   |   4500 |          8
     7 | develop   |   4200 |          9
     2 | personnel |   3900 |         10
     5 | personnel |   3500 |         11
(11 rows)
```

More Ranking

```
SELECT
    empno,
    depname,
    salary,
    row_number() OVER (
        ORDER BY salary DESC NULLS LAST
    ),
    rank() OVER (
        ORDER BY salary DESC NULLS LAST
    ),
    dense_rank() OVER (
        ORDER BY salary DESC NULLS LAST
    )
FROM
    empsalary
ORDER BY salary DESC;
```

More Ranking

```
 empno |  depname  | salary | row_number | rank | dense_rank
-------+-----------+--------+------------+------+------------
     8 | develop   |   6000 |          1 |    1 |          1
     6 | sales     |   5500 |          2 |    2 |          2
    10 | develop   |   5200 |          3 |    3 |          3
    11 | develop   |   5200 |          4 |    3 |          3
     1 | sales     |   5000 |          5 |    5 |          4
     3 | sales     |   4800 |          6 |    6 |          5
     4 | sales     |   4800 |          7 |    6 |          5
     9 | develop   |   4500 |          8 |    8 |          6
     7 | develop   |   4200 |          9 |    9 |          7
     2 | personnel |   3900 |         10 |   10 |          8
     5 | personnel |   3500 |         11 |   11 |          9
(11 rows)
```

PARTITIONing

```
SELECT
    empno,
    depname,
    salary,
    rank() OVER (
        PARTITION BY depname
        ORDER BY salary DESC NULLS LAST
    ) AS rank_in_dept,
    rank() OVER (
        ORDER BY salary DESC NULLS LAST
    ) AS global_rank
FROM
    empsalary;
```

PARTITIONing

```
 empno |  depname  | salary | rank_in_dept | global_rank
-------+-----------+--------+--------------+-------------
     8 | develop   |   6000 |            1 |           1
     6 | sales     |   5500 |            1 |           2
    10 | develop   |   5200 |            2 |           3
    11 | develop   |   5200 |            2 |           3
     1 | sales     |   5000 |            2 |           5
     4 | sales     |   4800 |            3 |           6
     3 | sales     |   4800 |            3 |           6
     9 | develop   |   4500 |            4 |           8
     7 | develop   |   4200 |            5 |           9
     2 | personnel |   3900 |            1 |          10
     5 | personnel |   3500 |            2 |          11
(11 rows)
```

Ranking on Aggregates

```
SELECT
    depname,
    sum(salary) AS total_salary,
    rank() OVER (
        ORDER BY sum(salary) DESC NULLS LAST
    ) AS rank_of_dept
FROM
    empsalary
GROUP BY depname;
```

Ranking on Aggregates

```
  depname  | total_salary | rank_of_dept
-----------+--------------+--------------
 develop   |        25100 |            1
 sales     |        20100 |            2
 personnel |         7400 |            3
(3 rows)
```

WINDOW

```
SELECT
    depname,
    empno,
    salary,
    SUM(salary) OVER w
FROM
    empsalary WINDOW w AS (
        PARTITION BY depname
        ORDER BY salary, empno
    )
;
```

WINDOW

```
SELECT
    depname,
    empno,
    salary,
    SUM(salary) OVER w
FROM
    empsalary WINDOW w AS (
        PARTITION BY depname
        ORDER BY salary, empno
    )
;
```

WINDOW

```
  depname  | empno | salary |  sum
-----------+-------+--------+-------
 develop   |     7 |   4200 | 25100
 develop   |     9 |   4500 | 25100
 develop   |    10 |   5200 | 25100
 develop   |    11 |   5200 | 25100
 develop   |     8 |   6000 | 25100
 personnel |     5 |   3500 |  7400
 personnel |     2 |   3900 |  7400
 sales     |     3 |   4800 | 20100
 sales     |     4 |   4800 | 20100
 sales     |     1 |   5000 | 20100
 sales     |     6 |   5500 | 20100
(11 rows)
```

LEAD() and LAG()

Prospects for Advancement

```
SELECT
    depname,
    salary,
    lag(salary,1) OVER (
        PARTITION BY depname
        ORDER BY salary desc
    ) - salary AS "delta"
FROM empsalary;
```

Prospects for Advancement

```
SELECT
    depname,
    salary,
    lag(salary,1) OVER (
        PARTITION BY depname
        ORDER BY salary desc
    ) - salary AS "delta"
FROM empsalary;
```

Prospects for Advancement

```
SELECT
    depname,
    salary,
    lag(salary,1) OVER (
        PARTITION BY depname
        ORDER BY salary desc
    ) - salary AS "delta"
FROM empsalary;
```

Prospects for Advancement

```
SELECT
    depname,
    salary,
    lag(salary,1) OVER (
        PARTITION BY depname
        ORDER BY salary desc
    ) - salary AS "delta"
FROM empsalary;
```

Prospects for Advancement

```
SELECT
    depname,
    salary,
    lag(salary,1) OVER (
        PARTITION BY depname
        ORDER BY salary desc
    ) - salary AS "delta"
FROM empsalary;
```

Prospects for Advancement

```
  depname  | salary | delta
-----------+--------+-------
 develop   |   6000 |
 develop   |   5200 |   800
 develop   |   5200 |     0
 develop   |   4500 |   700
 develop   |   4200 |   300
 personnel |   3900 |
 personnel |   3500 |   400
 sales     |   5500 |
 sales     |   5000 |   500
 sales     |   4800 |   200
 sales     |   4800 |     0
(11 rows)
```

Prospects for Advancement

```
WITH raises AS (
    SELECT
        depname,
        salary,
        lag(salary,1) OVER (
            PARTITION BY depname
            ORDER BY salary desc
        ) AS raise_lag
    FROM empsalary
)
SELECT
    depname,
    salary,
    floor(
        100 *
        (raise_lag/salary::float - 1)
    ) AS "percent_raise"
FROM raises
WHERE raise_lag IS NOT NULL;
```

Prospects for Advancement

```
  depname  | salary | percent_raise
-----------+--------+---------------
 develop   |   5200 |            15
 develop   |   5200 |             0
 develop   |   4500 |            15
 develop   |   4200 |             7
 personnel |   3500 |            11
 sales     |   5000 |            10
 sales     |   4800 |             4
 sales     |   4800 |             0
(8 rows)
```

- Questions
- Comments
- Rocks

More!

http://developer.postgresql.org/pgdocs/postgres/tutorial-window.html

Thanks!