Astronomical Data Processing Using SciQL, an SQL Based Query Language for Array Data

Ying Zhang, Bart Scheers, Martin Kersten, Milena Ivanova, Niels Nes
CWI Amsterdam

ADASS XXI, Nov. 06-10, 2011, Paris, France

FrontPage | MonetDB

http://www.monetdb.org/

Google

Apple Yahoo! Google Maps YouTube Wikipedia News (20) Popular

DOWNLOAD NOW

features documentation developers

An Open-Source Database System

Why Not RDBMS?

- SQL is difficult
  - No appropriate array denotations
  - No functional complete operation set
- DBMSs are slow
  - Too much overhead
  - Size limitations (due to BLOB representations)
  - Existing foreign files
  - Scale
  - ...

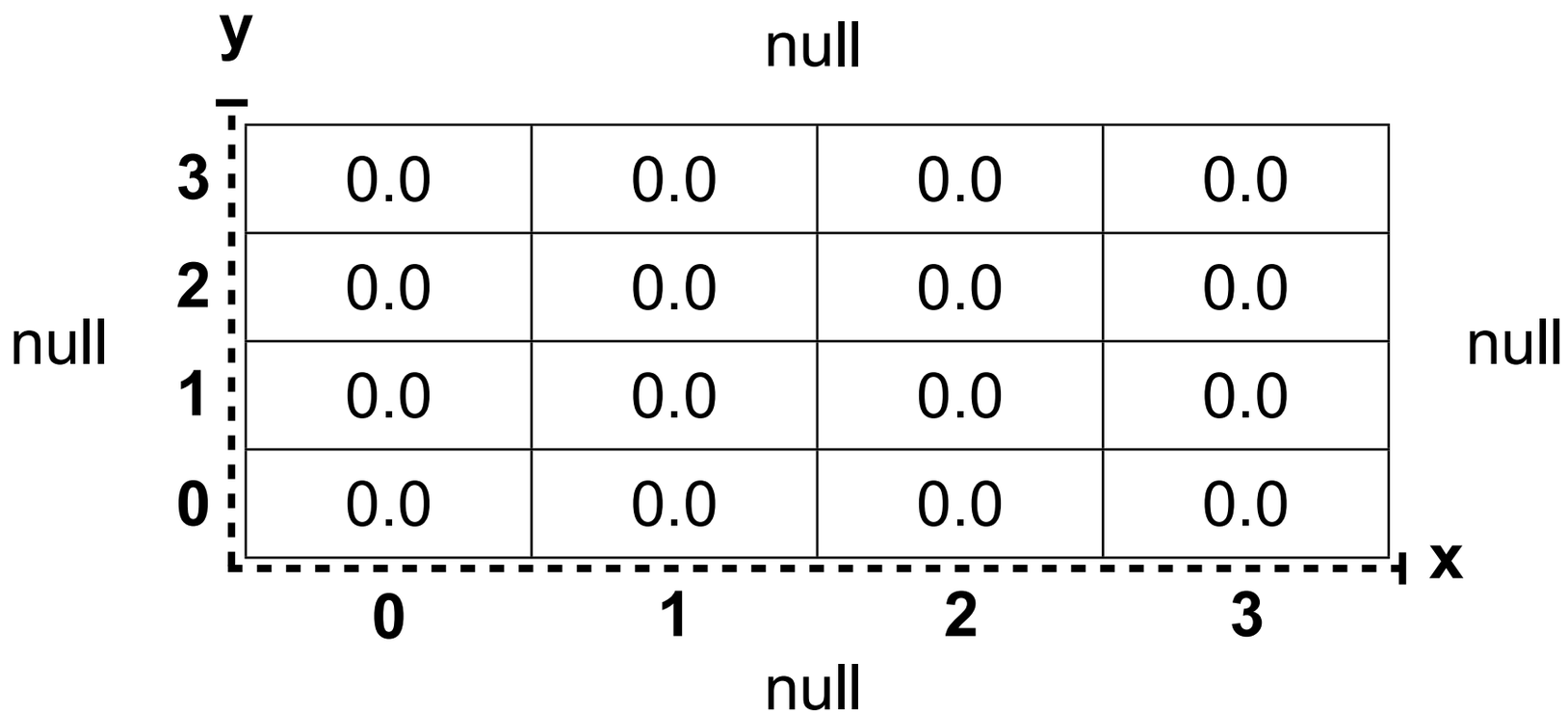
SciQL

- An array query language based on SQL:2003
- To lower the entrance fee to RDBMSs

- Distinguish features (Kersten et al. AD ’11; Zhang et al. IDEAS2011):
  - *Arrays and tables as first class citizens of DBMSs*
  - *Seamless integration of relational and array paradigms*
  - *Named dimensions with constraints*
  - *Flexible structure-based grouping*

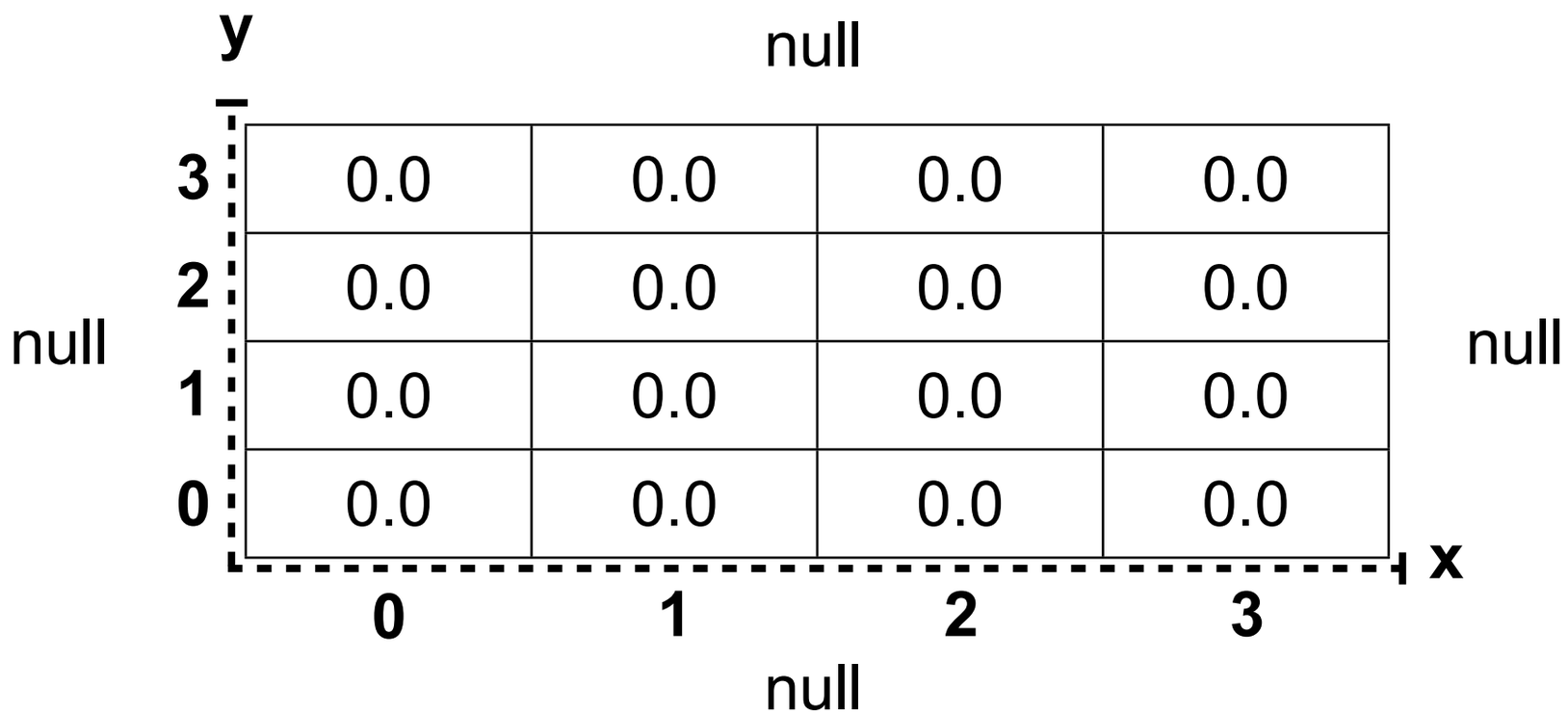
- LOFAR Transient Key Project use case

Array Definitions




```
CREATE ARRAY A1 (
  x INT DIMENSION [0:1:4], y INT DIMENSION [0:1:4],
  v FLOAT DEFAULT 0.0);
```

Array Definitions



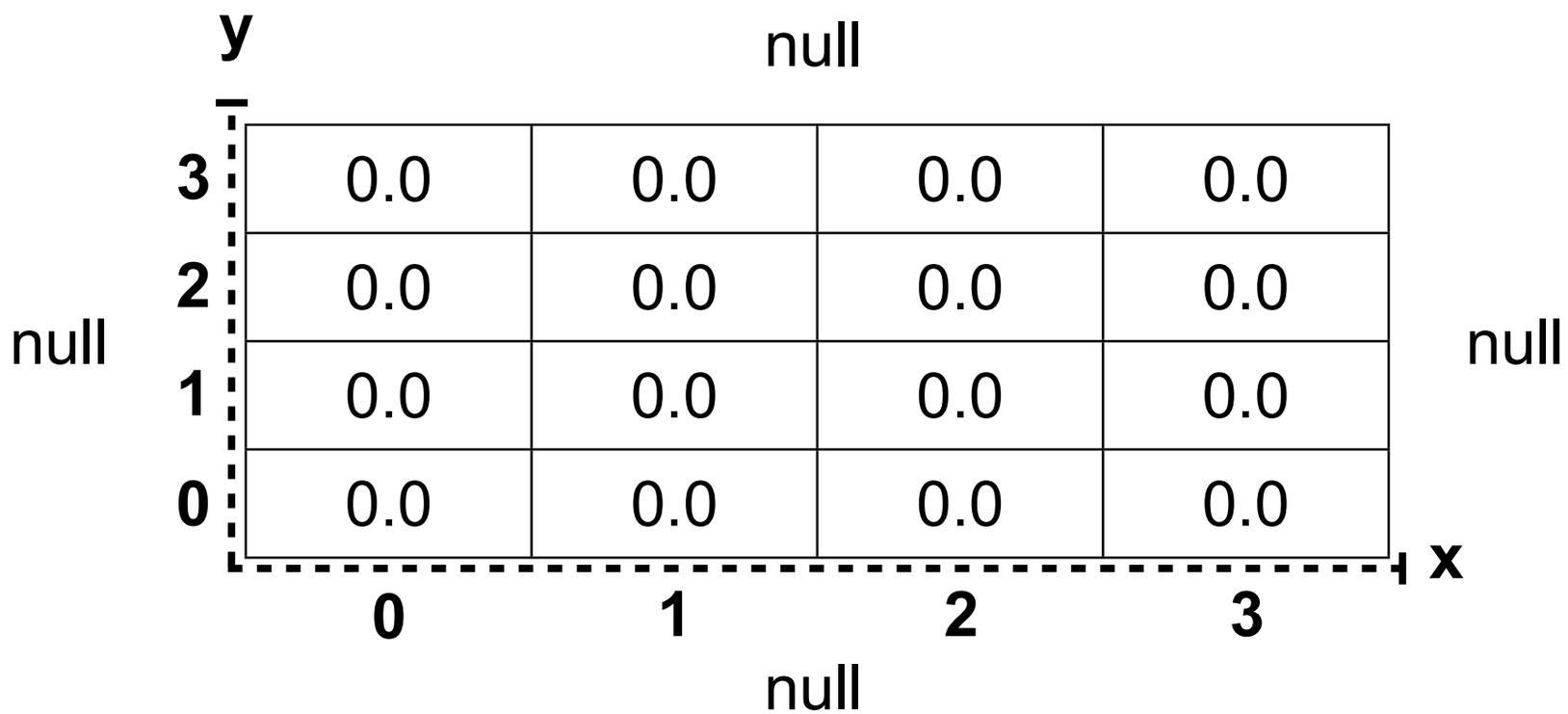
CREATE ARRAY A1 (

x INT **DIMENSION [0:1:4]**, y INT **DIMENSION [0:1:4]**,

v FLOAT DEFAULT 0.0);

dimensions,
any scalar data type

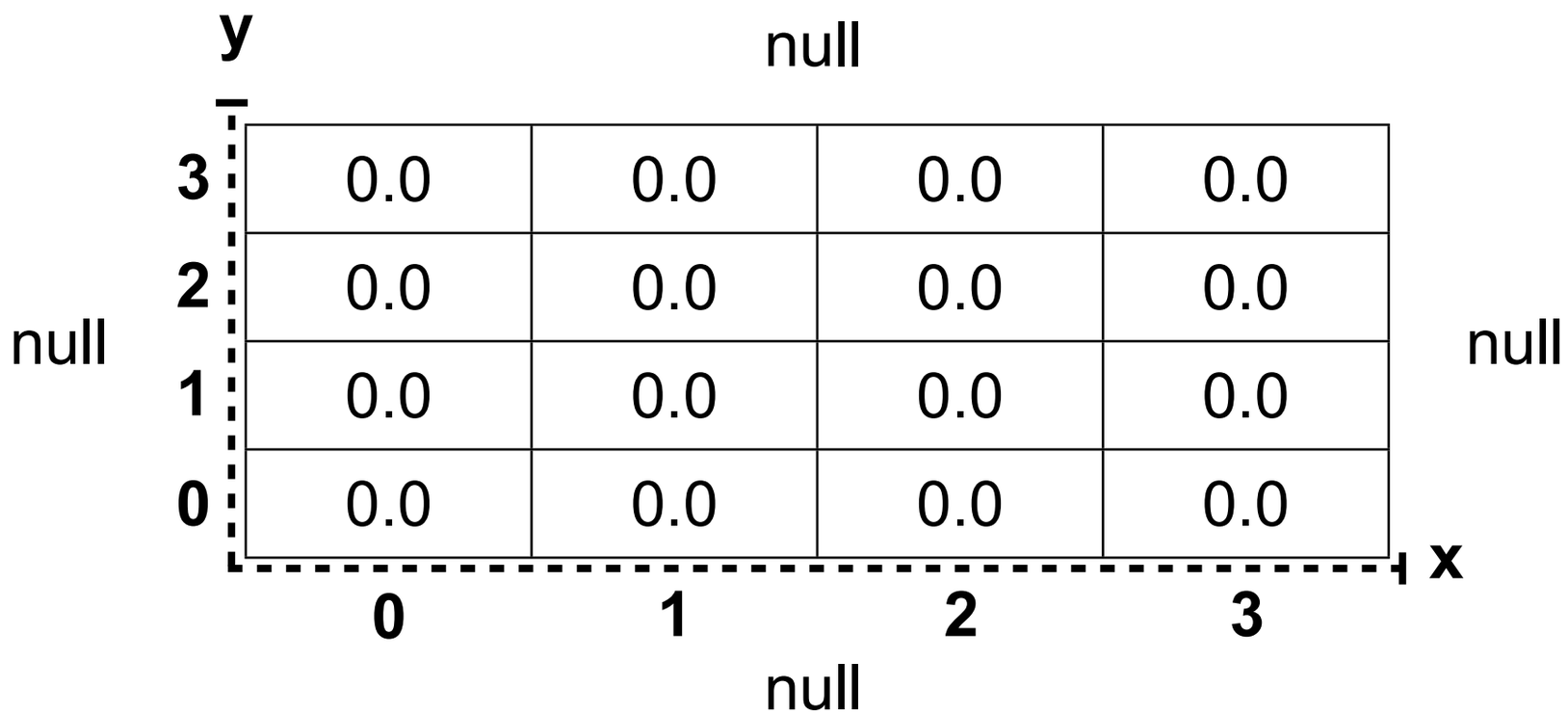
Array Definitions




```
CREATE ARRAY A1 (
  x INT DIMENSION [0:1:4], y INT DIMENSION [0:1:4],
  v FLOAT DEFAULT 0.0);
```

dimensional range:
[(start|∗) : (step|∗) : (stop|∗)]

Array Definitions

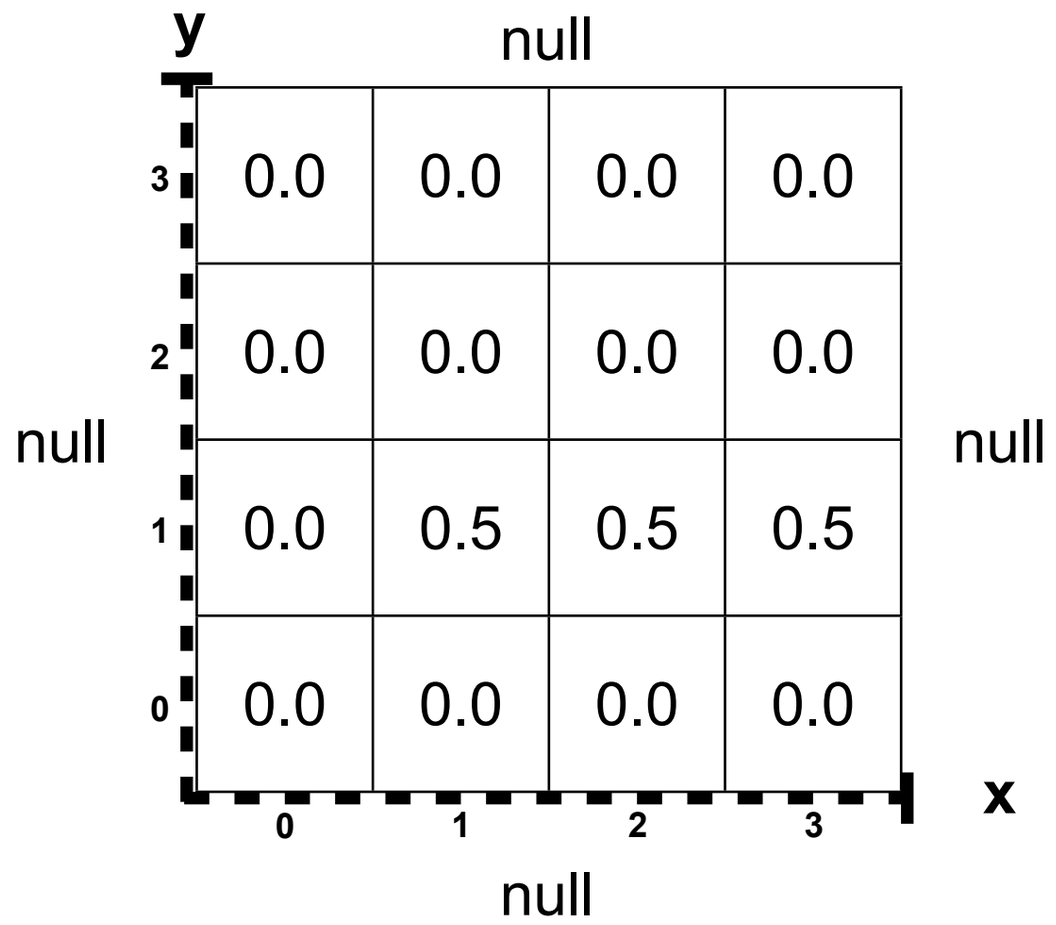



```
CREATE ARRAY A1 (
  x INT DIMENSION [0:1:4], y INT DIMENSION [0:1:4],
  v FLOAT DEFAULT 0.0);
```

cell values,
any column data type

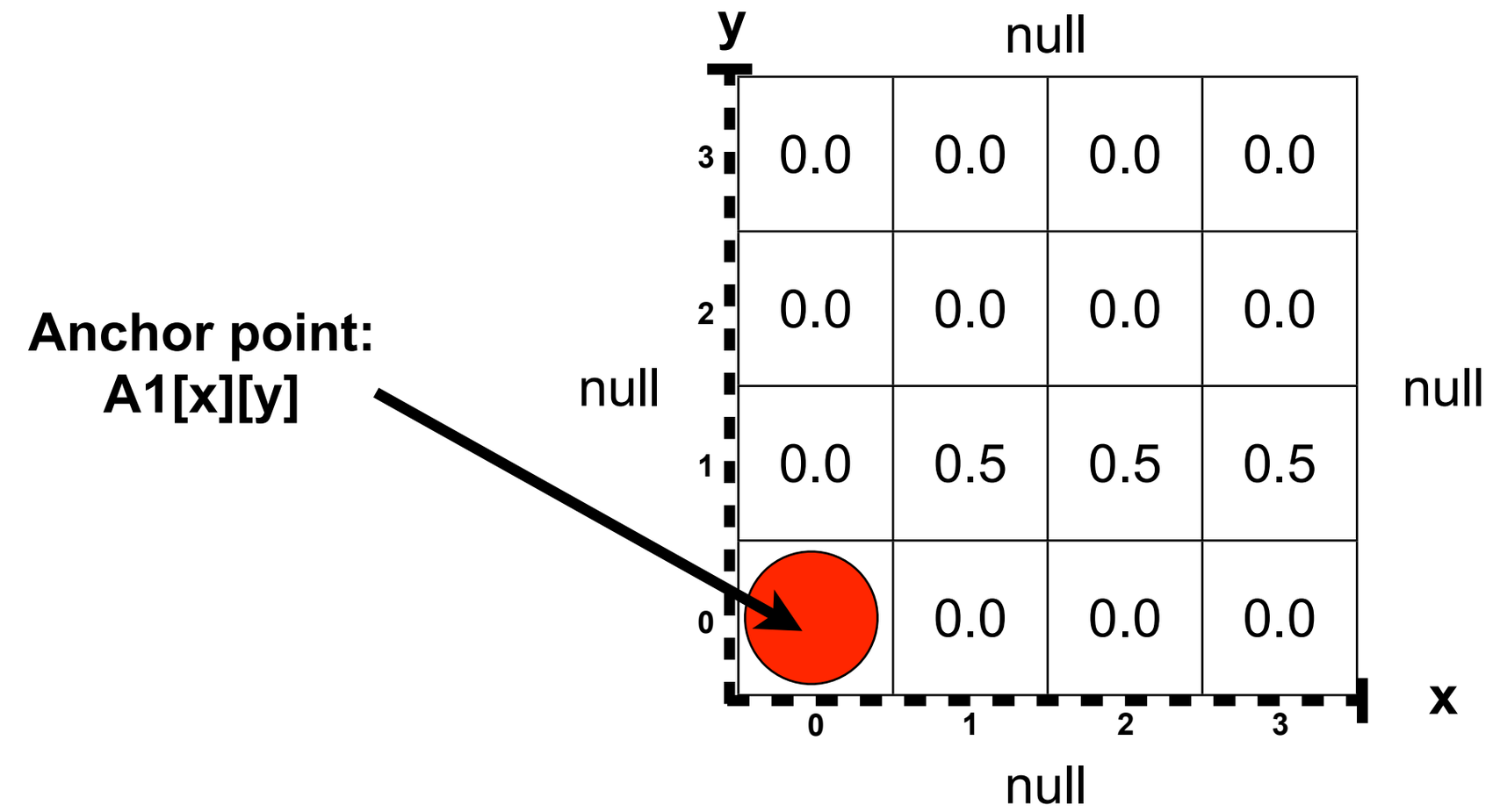
Array Tiling

SELECT [x], [y], **AVG(v)** FROM A1
GROUP BY A1[x:x+2][y:y+2];



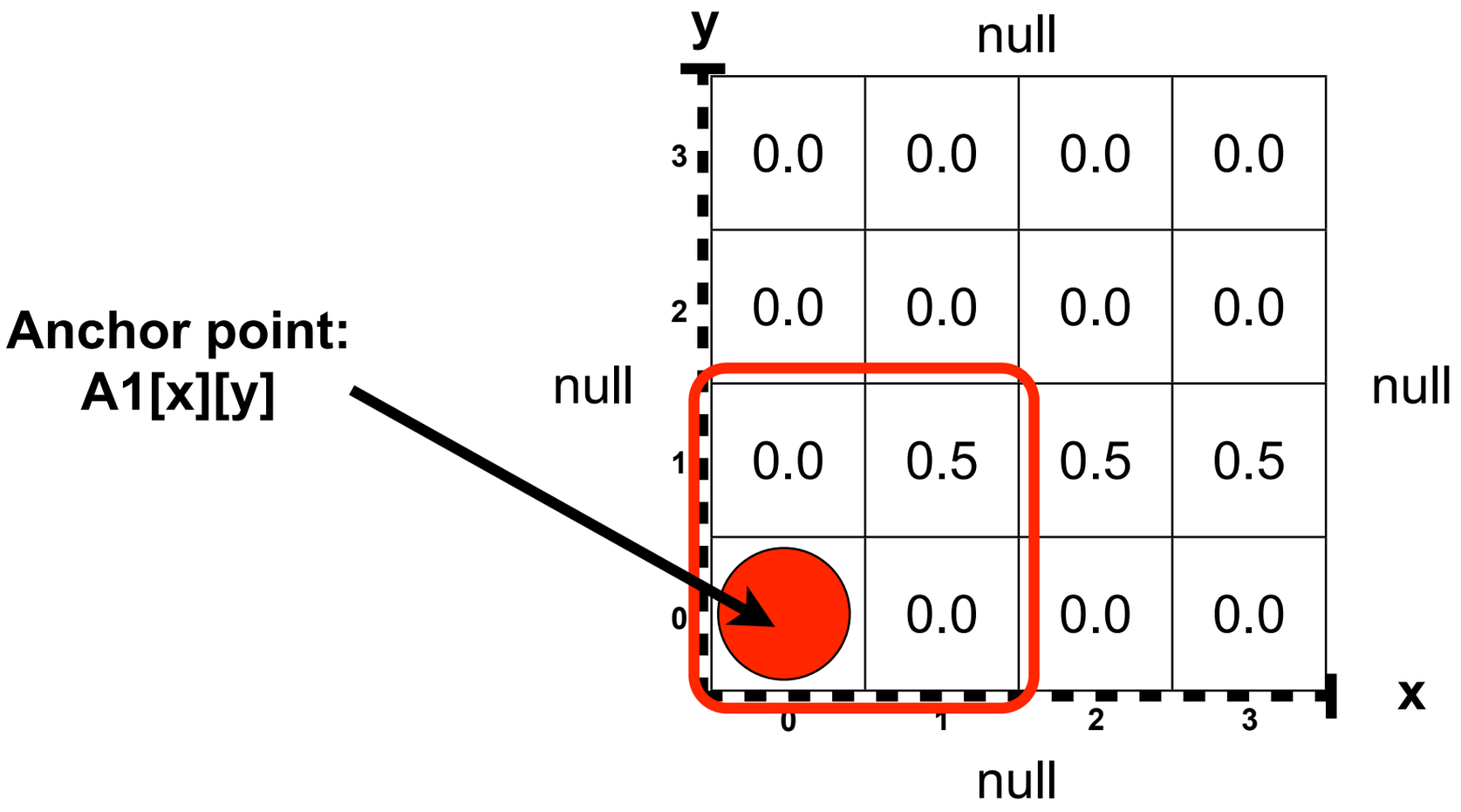
Array Tiling

SELECT [x], [y], **AVG(v)** FROM A1
GROUP BY A1[x:x+2][y:y+2];



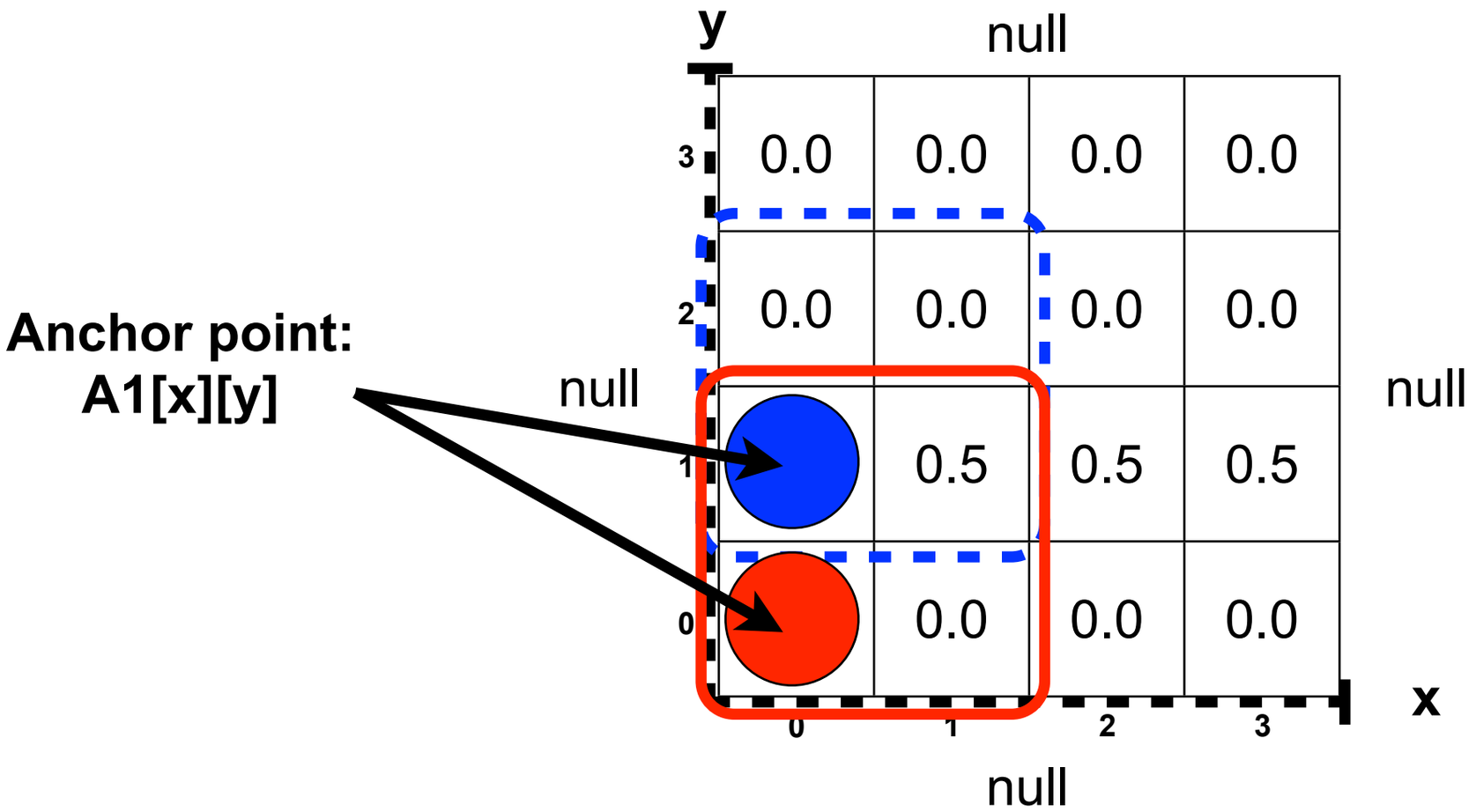
Array Tiling

SELECT [x], [y], **AVG(v)** FROM A1
GROUP BY A1[x:x+2][y:y+2];



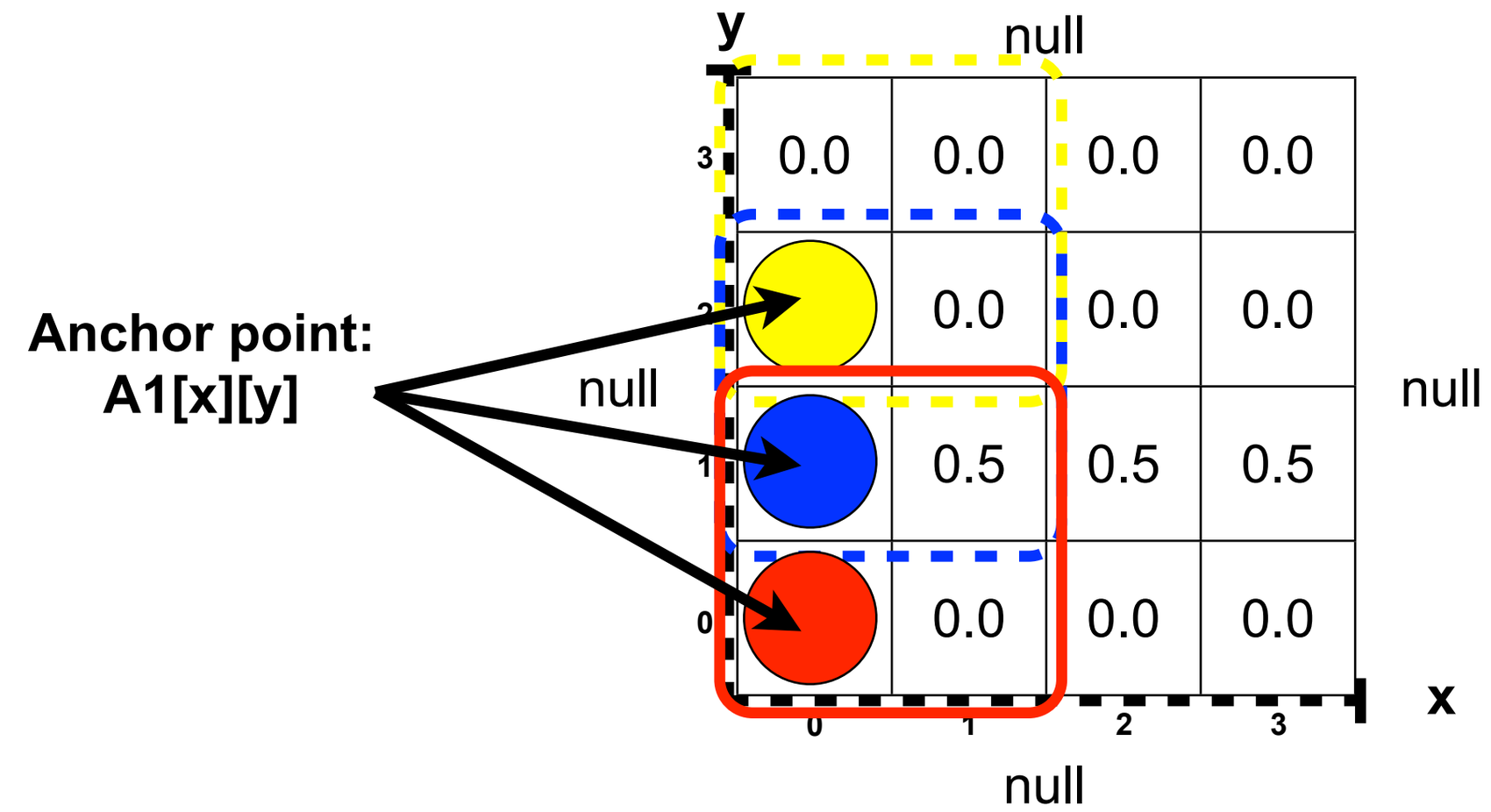
Array Tiling

SELECT [x], [y], **AVG(v)** FROM A1
GROUP BY A1[x:x+2][y:y+2];



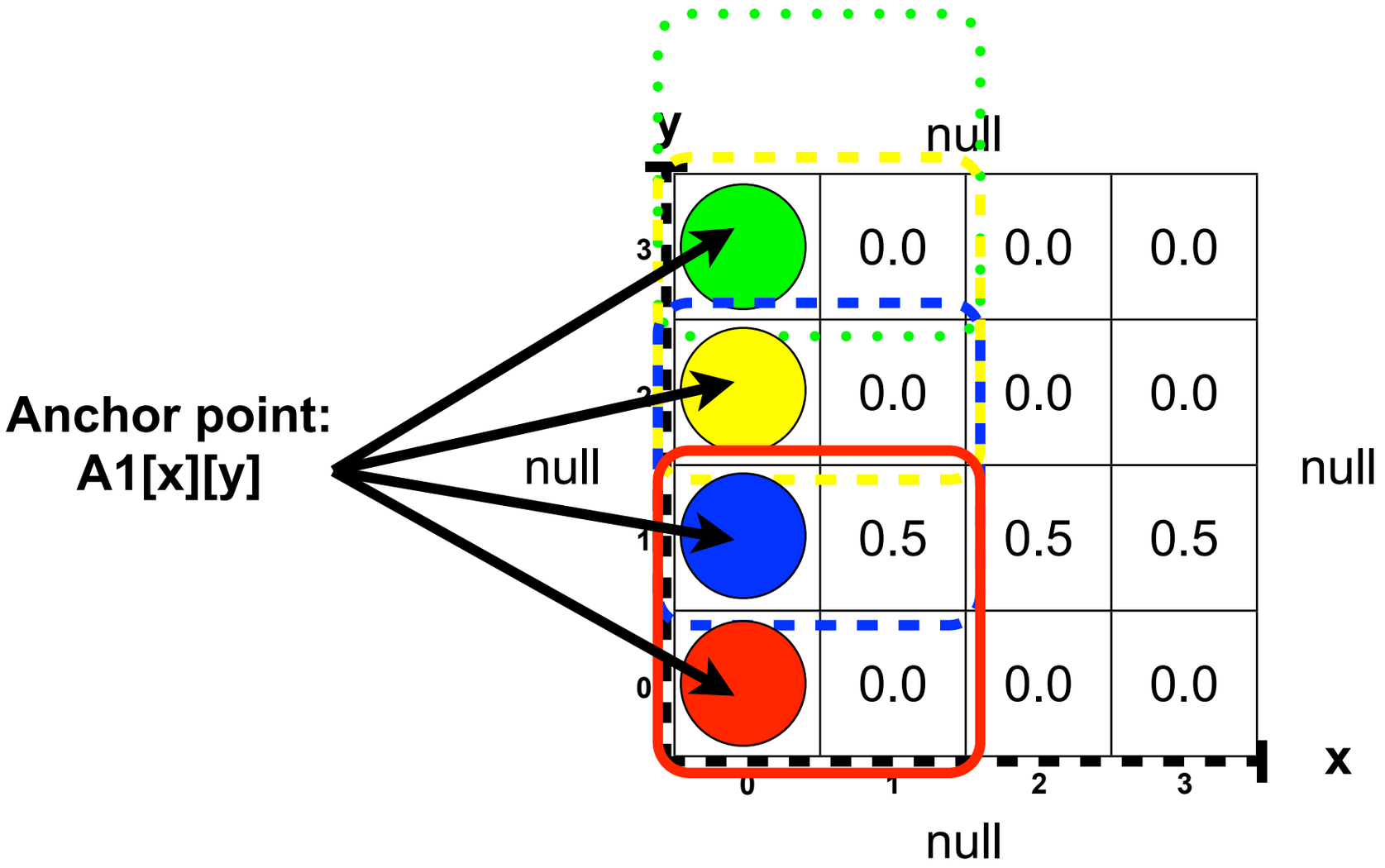
Array Tiling

SELECT [x], [y], **AVG(v)** FROM A1
GROUP BY A1[x:x+2][y:y+2];



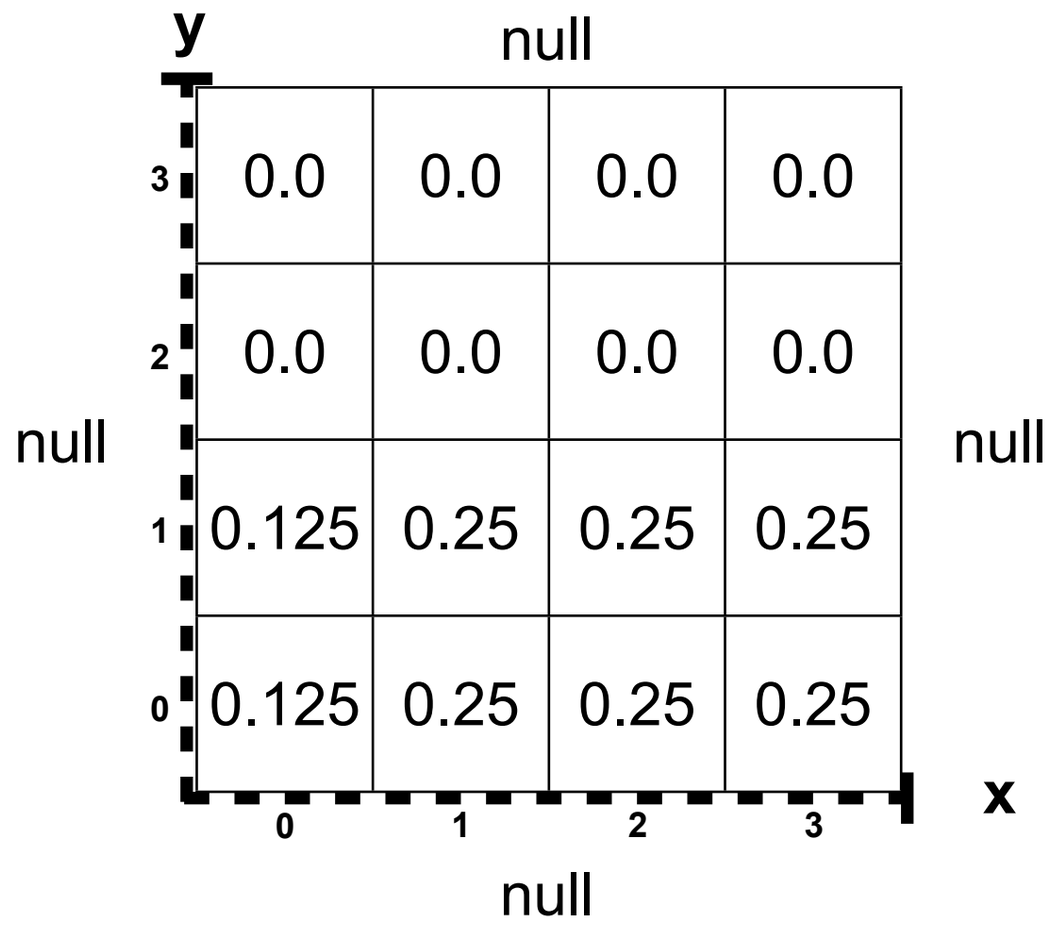
Array Tiling

SELECT [x], [y], **AVG(v)** FROM A1
GROUP BY A1[x:x+2][y:y+2];

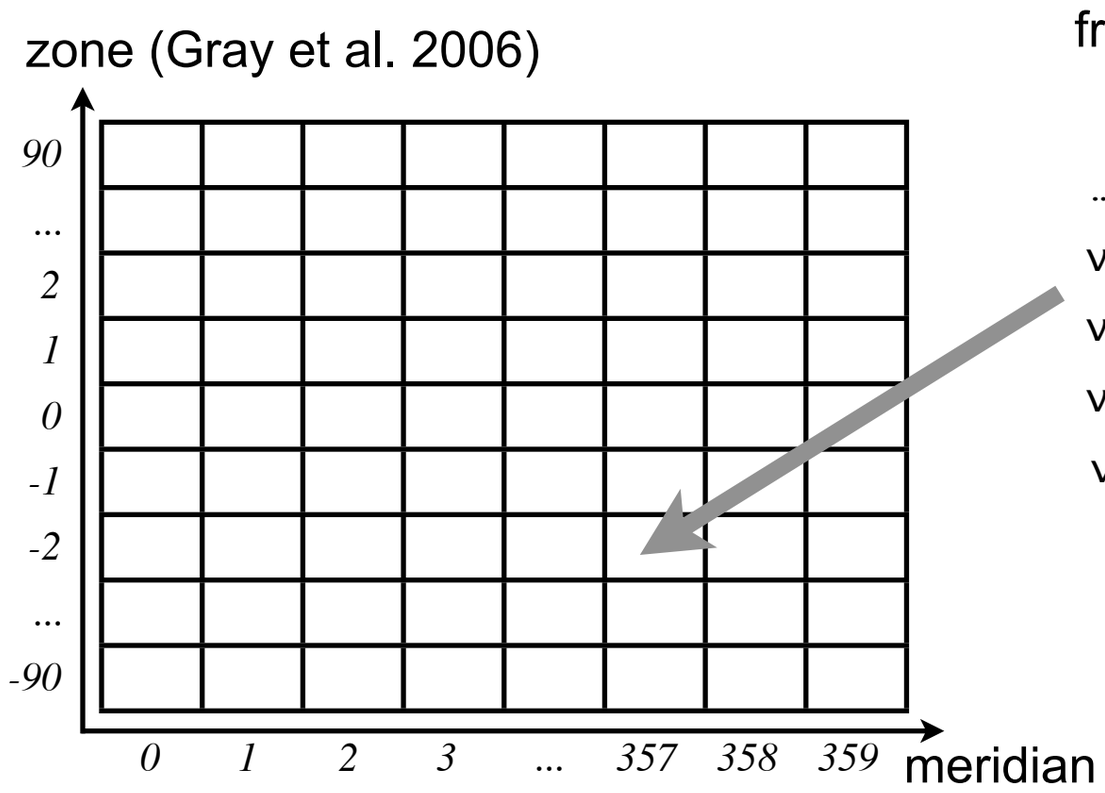


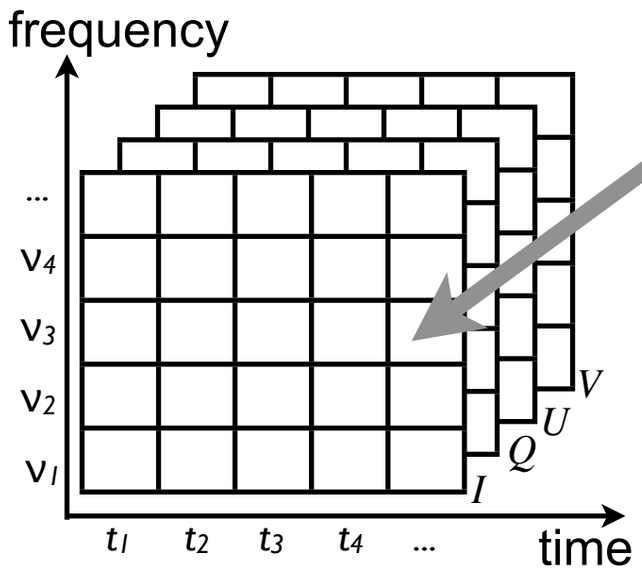
Array Tiling

SELECT [x], [y], **AVG(v)** FROM A1
GROUP BY A1[x:x+2][y:y+2];



LOFAR Catalogue

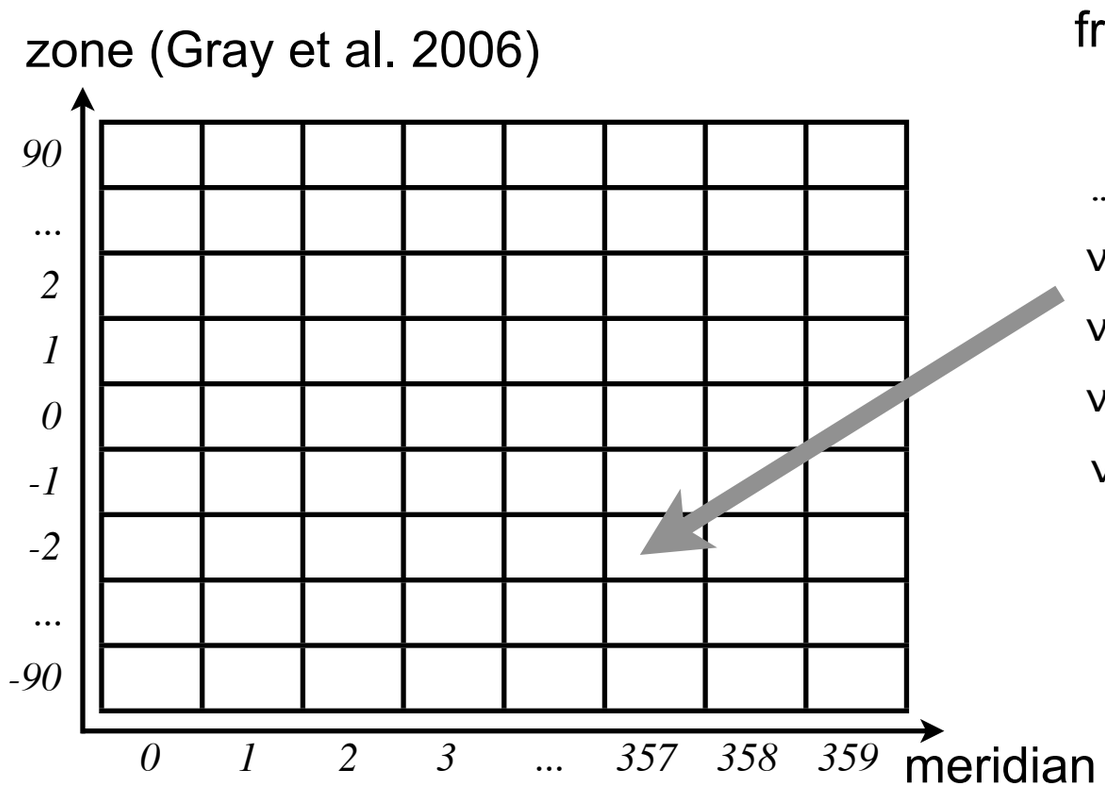


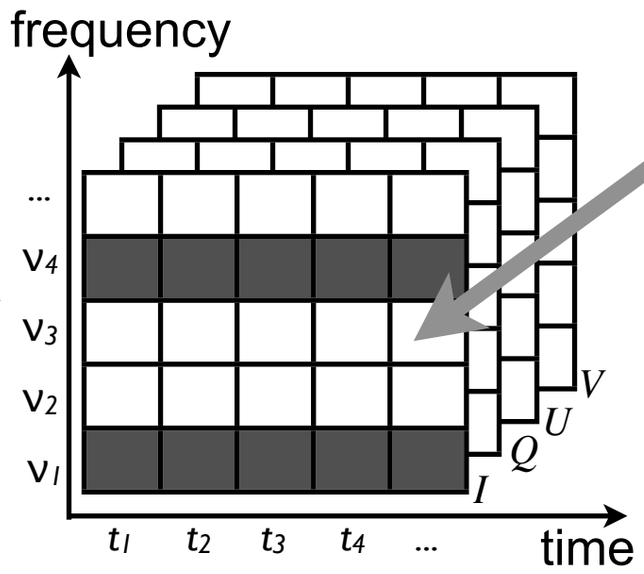



```
ra DOUBLE,
decl DOUBLE,
ra_err DOUBLE,
decl_err DOUBLE,
flux DOUBLE,
...
```

```
CREATE ARRAY LOFARsrc (
  zone INT DIMENSION[-90:1:91], mrdn INT DIMENSION[0:1:360],
  ts   TIMESTAMP DIMENSION,     freq INT DIMENSION[30:10:241],
  id   INT DIMENSION[0:1:*],    stks CHAR(1) DIMENSION
          CHECK(stks=`I' OR stks=`Q' OR stks=`U' OR stks=`V'),
  ra DOUBLE, decl DOUBLE, ra_err DOUBLE, decl_err DOUBLE,
  flux DOUBLE, ...);
```

LOFAR Use Case






```
ra DOUBLE,
decl DOUBLE,
ra_err DOUBLE,
decl_err DOUBLE,
flux DOUBLE,
...
```

Similarity of the flux of a LOFAR source at frequencies 30 MHz and 200 MHz

⇨ cross-correlation of two time series

Cross-Correlation

F

idx	0	1	2	3
val	4	3	6	2

G

idx	0	1	2
val	1	5	7

Cr

idx	-3	-2	-1	0	1	2
val						

Cross-Correlation

Cr.idx = -3

F [3 : 4]

G [0 : 1]

F

idx	0	1	2	3
val	4	3	6	2

G

idx	0	1	2
val	1	5	7

Cr

idx	-3	-2	-1	0	1	2
val	2					

Cross-Correlation

Cr.idx = -2

F

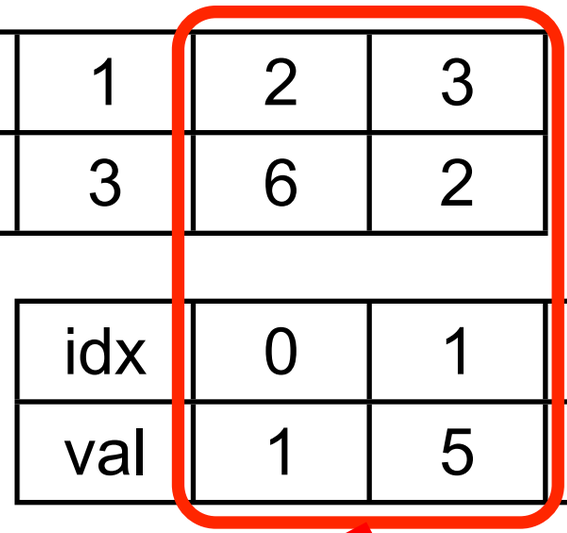
idx	0	1	2	3
val	4	3	6	2

F [2 : 4]

G

idx	0	1	2
val	1	5	7

G [0 : 2]



Cr

idx	-3	-2	-1	0	1	2
val	2	16				

Cross-Correlation

Cr.idx = -1

F

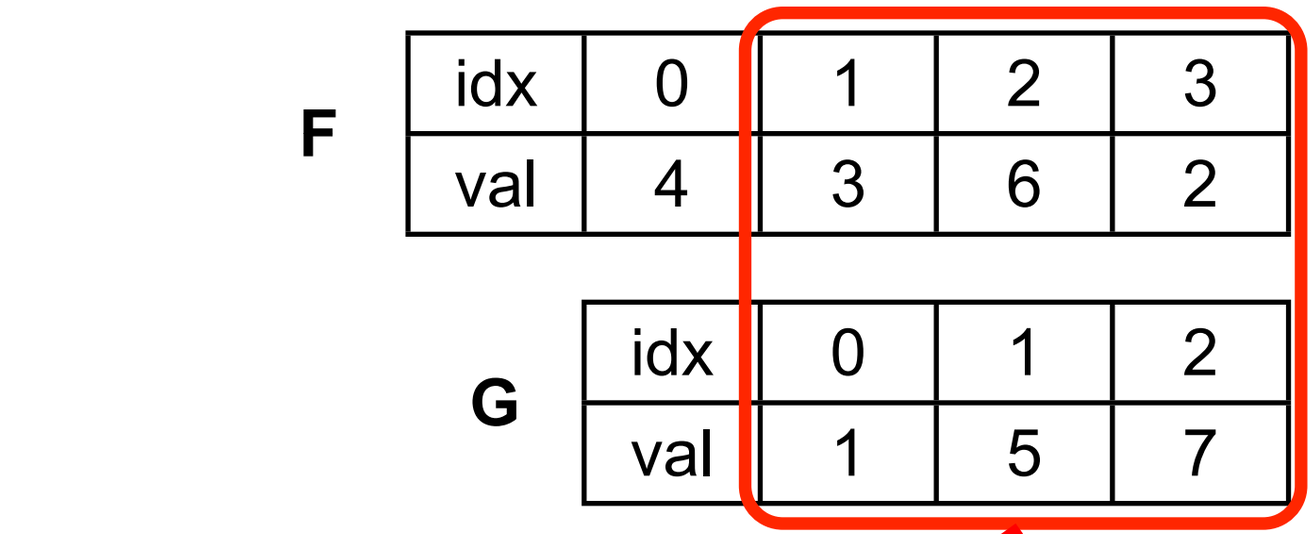
idx	0	1	2	3
val	4	3	6	2

F [1 : 4]

G

idx	0	1	2
val	1	5	7

G [0 : 3]



Cr

idx	-3	-2	-1	0	1	2
val	2	16	47			

Cross-Correlation

Cr.idx = 0

F

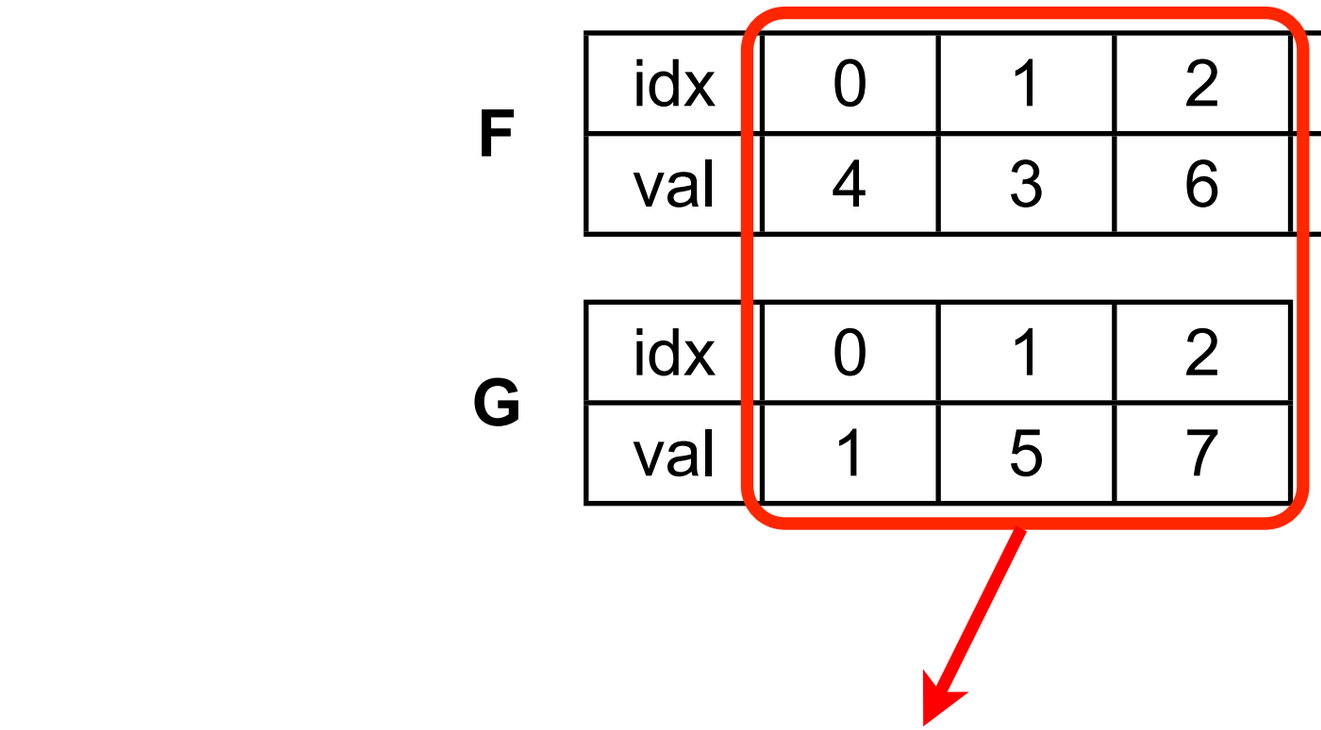
idx	0	1	2	3
val	4	3	6	2

F [0 : 3]

G

idx	0	1	2
val	1	5	7

G [0 : 3]



Cr

idx	-3	-2	-1	0	1	2
val	2	16	47	61		

Cross-Correlation

Cr.idx = 1

F

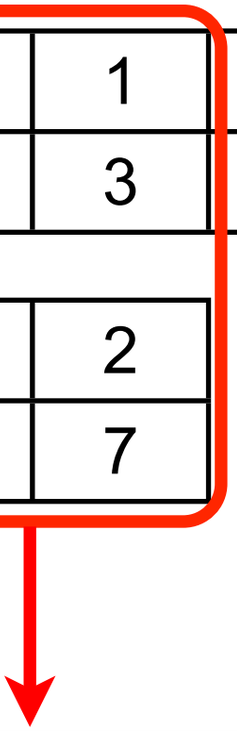
idx	0	1	2	3
val	4	3	6	2

F [0 : 2]

G

idx	0	1	2
val	1	5	7

G [1 : 3]



Cr

idx	-3	-2	-1	0	1	2
val	2	16	47	61	41	

Cross-Correlation

Cr.idx = 2

F

idx	0	1	2	3
val	4	3	6	2

F [0 : 1]

G

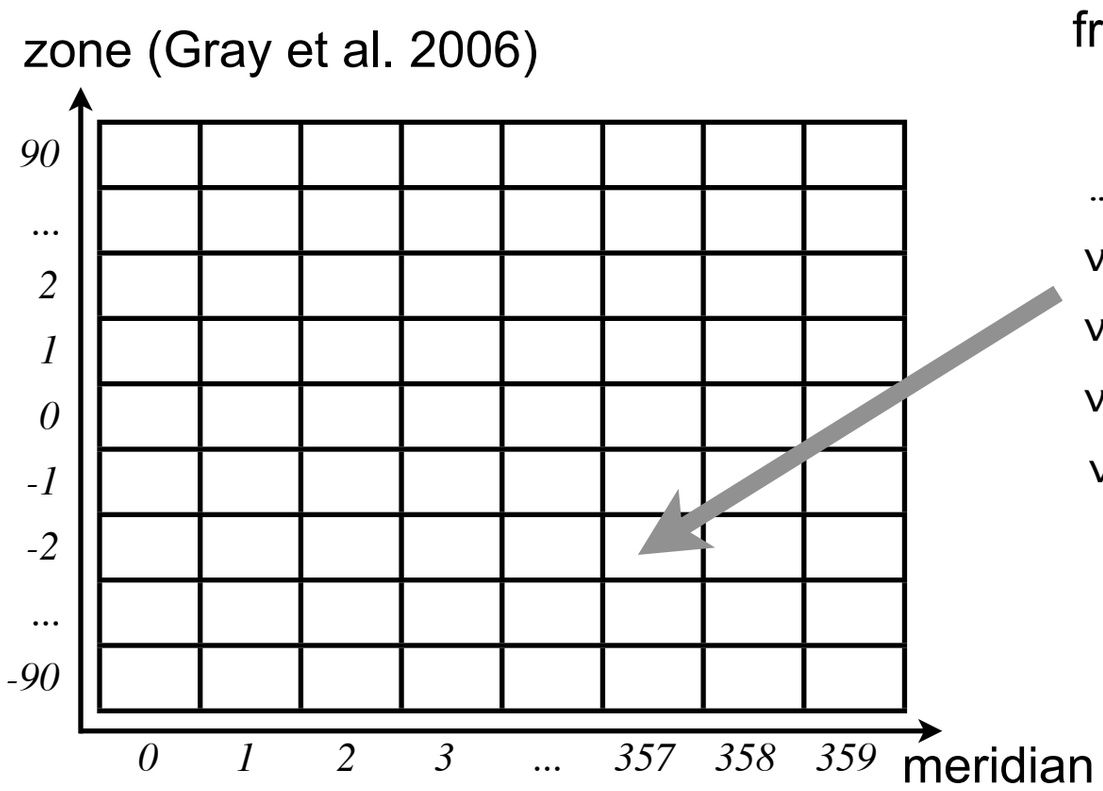
idx	0	1	2
val	1	5	7

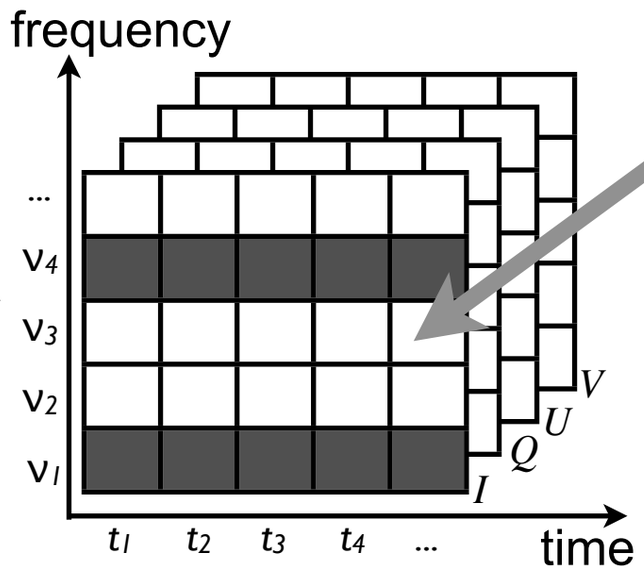
G [2 : 3]

Cr

idx	-3	-2	-1	0	1	2
val	2	16	47	61	41	28

LOFAR Use Case





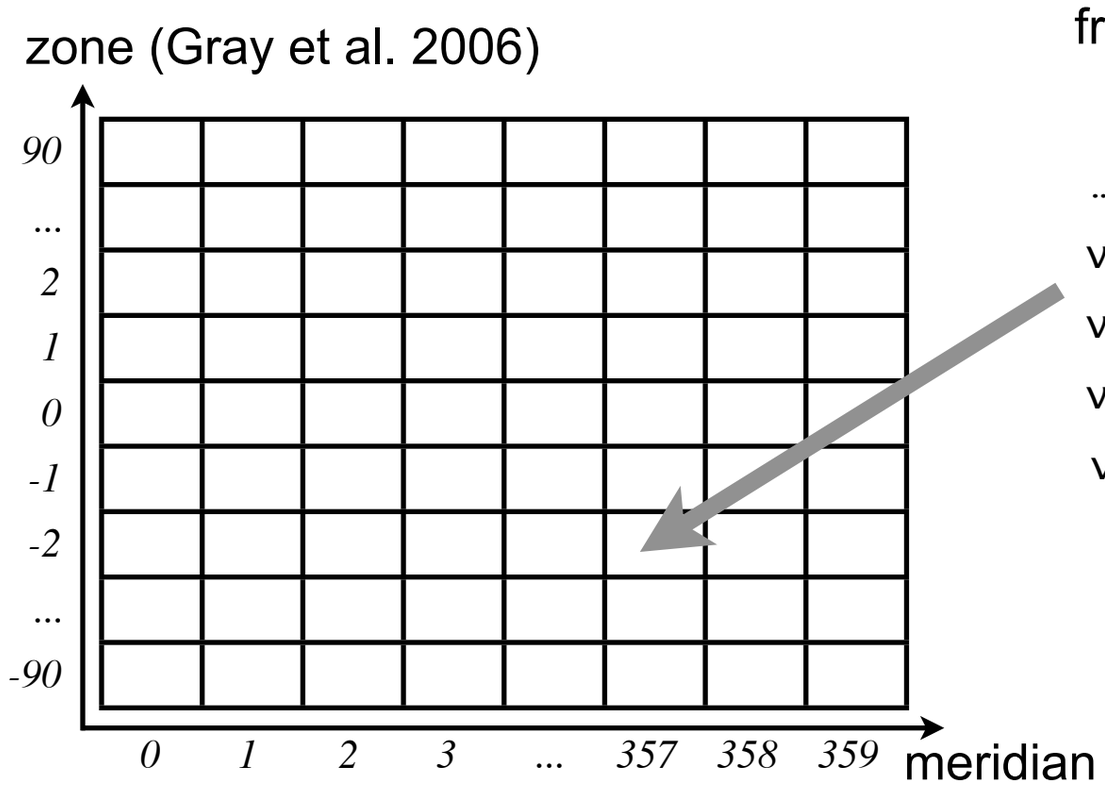

```
ra DOUBLE,
decl DOUBLE,
ra_err DOUBLE,
decl_err DOUBLE,
flux DOUBLE,
...
```

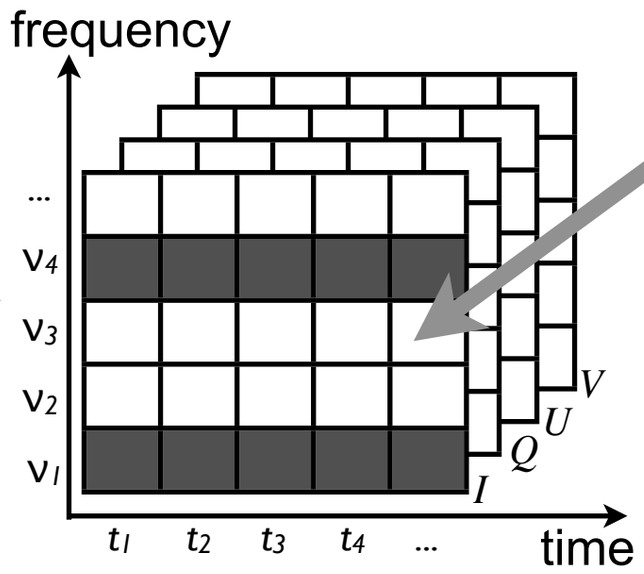
```
DECLARE fcnt INT, gcnt INT;
SET fcnt = SELECT COUNT(*) FROM LOFARsrc[*][*][*][30][11]['I'];
SET gcnt = SELECT COUNT(*) FROM LOFARsrc[*][*][*][200][11]['I'];

CREATE ARRAY VIEW F (idx INT DIMENSION[0:1:fcnt], flux DOUBLE DEFAULT 0.0) AS SELECT flux FROM
   LOFARsrc[*][*][*][30][11]['I'];
CREATE ARRAY VIEW G (idx INT DIMENSION[0:1:gcnt], val DOUBLE DEFAULT 0.0) AS SELECT flux FROM
   LOFARsrc[*][*][*][200][11]['I'];

CREATE ARRAY CrCorr30_200 (idx INT DIMENSION[-fcnt+1:1:gcnt], val DOUBLE DEFAULT 0.0);
INSERT INTO CrCorr SELECT SUM(F.flux * G.flux) FROM F, G, CrCorr30_200 AS C
  GROUP BY F[MAX(0, -C.idx) : MIN(fcnt, gcnt-C.idx)], G[MAX(0, C.idx) : MIN(gcnt, fcnt+C.idx)];
```

LOFAR Use Case






```
ra DOUBLE,
decl DOUBLE,
ra_err DOUBLE,
decl_err DOUBLE,
flux DOUBLE,
...
```

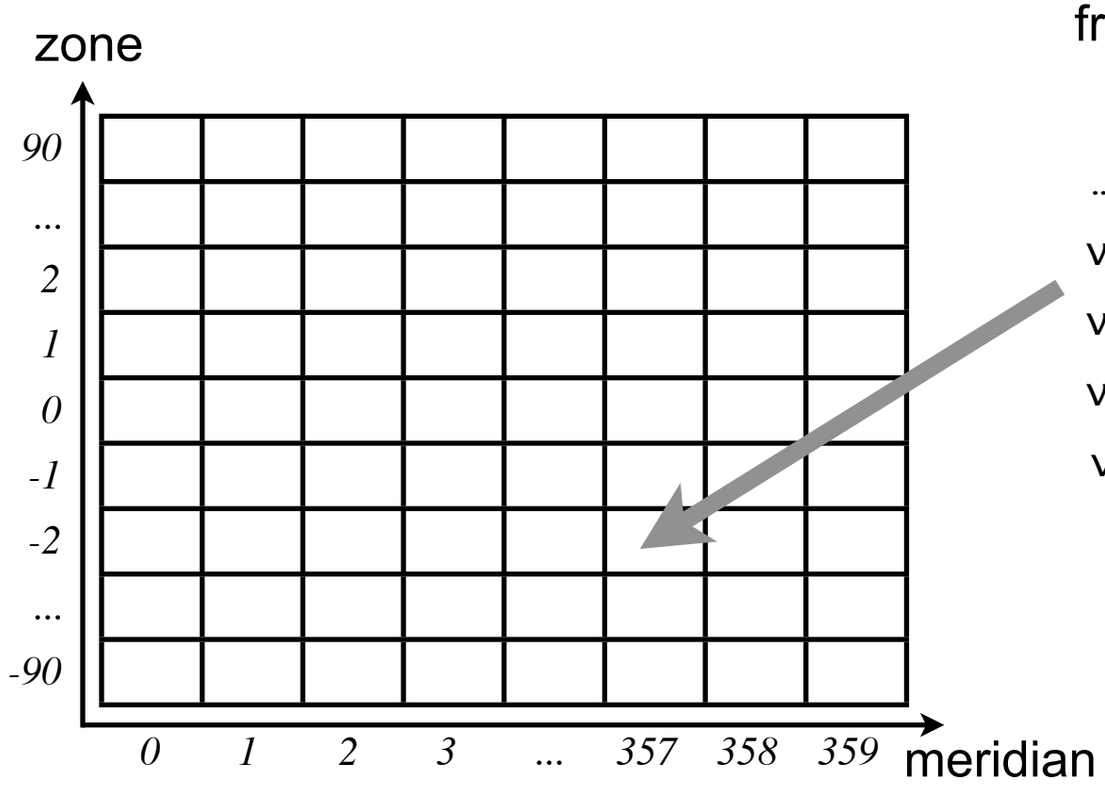
retrieve the time series

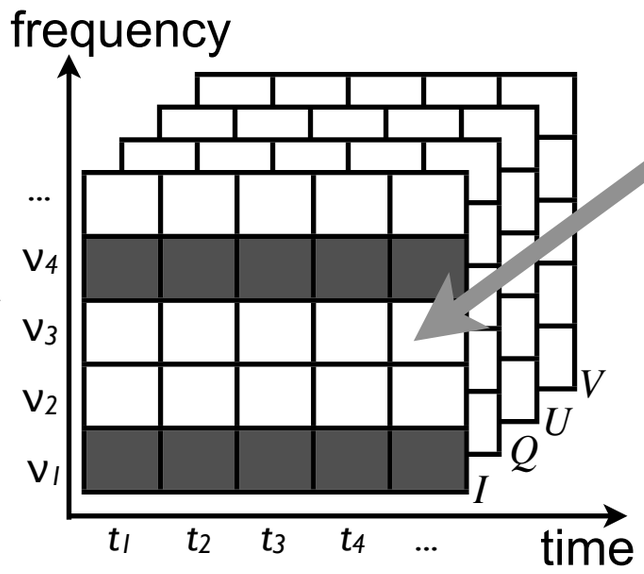
```
DECLARE fcnt INT, gcnt INT;
SET fcnt = SELECT COUNT(*) FROM LOFARsrc[*][*][*][30][11]['I'];
SET gcnt = SELECT COUNT(*) FROM LOFARsrc[*][*][*][200][11]['I'];

CREATE ARRAY VIEW F (idx INT DIMENSION[0:1:fcnt], flux DOUBLE DEFAULT 0.0) AS SELECT flux FROM
   LOFARsrc[*][*][*][30][11]['I'];
CREATE ARRAY VIEW G (idx INT DIMENSION[0:1:gcnt], val DOUBLE DEFAULT 0.0) AS SELECT flux FROM
   LOFARsrc[*][*][*][200][11]['I'];

CREATE ARRAY CrCorr30_200 (idx INT DIMENSION[-fcnt+1:1:gcnt], val DOUBLE DEFAULT 0.0);
INSERT INTO CrCorr SELECT SUM(F.flux * G.flux) FROM F, G, CrCorr30_200 AS C
  GROUP BY F[MAX(0, -C.idx) : MIN(fcnt, gcnt-C.idx)], G[MAX(0, C.idx) : MIN(gcnt, fcnt+C.idx)];
```

LOFAR Use Case






```
ra DOUBLE,
decl DOUBLE,
ra_err DOUBLE,
decl_err DOUBLE,
flux DOUBLE,
...
```

dynamic grouping for every iteration

```
DECLARE fcnt INT, gcnt INT;
SET fcnt = SELECT COUNT(*) FROM LOFARsrc[*][*][*][30][11]['I'];
SET gcnt = SELECT COUNT(*) FROM LOFARsrc[*][*][*][200][11]['I'];

CREATE ARRAY VIEW F (idx INT DIMENSION[0:1:fcnt], flux DOUBLE DEFAULT 0.0) AS SELECT flux FROM
   LOFARsrc[*][*][*][30][11]['I'];
CREATE ARRAY VIEW G (idx INT DIMENSION[0:1:gcnt], val DOUBLE DEFAULT 0.0) AS SELECT flux FROM
   LOFARsrc[*][*][*][200][11]['I'];

CREATE ARRAY CrCorr30_200 (idx INT DIMENSION[-fcnt+1:1:gcnt], val DOUBLE DEFAULT 0.0);
INSERT INTO CrCorr SELECT SUM(F.flux * G.flux) FROM F, G, CrCorr30_200 AS C
  GROUP BY F[MAX(0, -C.idx) : MIN(fcnt, gcnt-C.idx)], G[MAX(0, C.idx) : MIN(gcnt, fcnt+C.idx)];
```

Conclusion

- SciQL: a novel query language for scientific data
  - A symbiosis of relational and array paradigm
- Simplifies expression of complex scientific algorithms
- Leave optimisation to DBMS kernel
- *Opens opportunities to enhance scientific data mining*
- Under active implementation

www.scilens.org

www.monetdb.org