

## Consulta primera: Diagnóstico de duración en días de los contratos por ciudad

```
SELECT

    c.descripcion AS ciudad,

    MIN(p.duracion) AS duracion_minima,

    ROUND(AVG(p.duracion), 2) AS duracion_promedio,

    MAX(p.duracion) AS duracion_maxima

FROM Procesos p

JOIN Entidades e ON e.nit_entidad = p.nit_entidad

JOIN Ciudades c ON c.id = e.ciudad_id

GROUP BY c.descripcion

ORDER BY c.descripcion;
```

PLAN DE EJECUCION:

Operation	Params	Rows	Total Cost	Raw Desc
Select (SORT ORDER BY)		1	279.0	cpu_cost = 133910362, io_cost = 276
Order By (SORT ORDER BY)		1	279.0	cpu_cost = 133910362, io_cost = 276
Hash Join		1	278.0	cpu_cost = 91432085, io_cost = 276
Nested Loops		1	278.0	cpu_cost = 91432085, io_cost = 276
Nested Loops		1	278.0	cpu_cost = 91432085, io_cost = 276
Unknown (STATISTICS COLLECTOR)		1	277.0	cpu_cost = null, io_cost = null
Access (VIEW)		1	277.0	cpu_cost = 91422894, io_cost = 275
Group By (HASH GROUP BY)		1	277.0	cpu_cost = 91422894, io_cost = 275
Hash Join		1	278.0	cpu_cost = 48944817, io_cost = 275
Full Scan (TABLE ACCESS FULL)	table: ENTIDADES;	1	3.0	cpu_cost = 35907, io_cost = 3
Full Scan (TABLE ACCESS FULL)	table: PROCESOS;	111317	273.0	cpu_cost = 37777160, io_cost = 272
Unique Index Scan (INDEX UNIQUE SCAN)	index: SYS_CO08577;	1	0.0	cpu_cost = 1900, io_cost = 0
Index Scan (TABLE ACCESS BY INDEX ROWID)	table: CIUDADES;	1	1.0	cpu_cost = 9191, io_cost = 1
Full Scan (TABLE ACCESS FULL)	table: CIUDADES;	1	1.0	cpu_cost = 9191, io_cost = 1

Resultados: result\_1.csv

Posibles mejoras:

```
CREATE INDEX idx_procesos_nit_entidad ON Procesos(nit_entidad);
CREATE INDEX idx_entidades_ciudad_id ON Entidades(ciudad_id);
```

```
CREATE INDEX idx_procesos_duracion ON Procesos(duracion);
```

```
-- Índice compuesto para mejor rendimiento en JOINS
```

```
CREATE INDEX idx_entidades_nit_ciudad ON Entidades (nit_entidad, ciudad_id);
```

Operation	Params	Rows	Total Cost	Raw Desc
Order By (SORT ORDER BY)		1	115.0	cpu_cost = 85867739, io_cost = 113
Hash Join		1	115.0	cpu_cost = 85867739, io_cost = 113
Nested Loops		1	114.0	cpu_cost = 43389462, io_cost = 113
Nested Loops		1	114.0	cpu_cost = 43389462, io_cost = 113
Unknown (STATISTICS COLLECTOR)		1	114.0	cpu_cost = 43389462, io_cost = 113
Access (VIEW)		1	113.0	cpu_cost = null, io_cost = null
Group By (HASH GROUP BY)		1	113.0	cpu_cost = 43380271, io_cost = 112
Hash Join		1	112.0	cpu_cost = 901994, io_cost = 112
Nested Loops		1	112.0	cpu_cost = 901994, io_cost = 112
Nested Loops		154	112.0	cpu_cost = 901994, io_cost = 112
Unknown (STATISTICS COLLECTOR)		1	3.0	cpu_cost = null, io_cost = null
Full Scan (TABLE ACCESS FULL)	table: ENTIDADES;	1	3.0	cpu_cost = 35607, io_cost = 3
Index Scan (INDEX RANGE SCAN)	index: IDX_PROCESOS_NIT_ENTIDAD;	154	1.0	cpu_cost = 45971, io_cost = 1
Index Scan (TABLE ACCESS BY INDEX F)	table: PROCESOS;	154	109.0	cpu_cost = 866387, io_cost = 109
Full Scan (TABLE ACCESS FULL)	table: PROCESOS;	154	109.0	cpu_cost = 866387, io_cost = 109
Unique Index Scan (INDEX UNIQUE SCAN)	index: SYS_C008577;	1	0.0	cpu_cost = 1900, io_cost = 0
Index Scan (TABLE ACCESS BY INDEX ROWID)	table: CIUDADES;	1	1.0	cpu_cost = 9191, io_cost = 1
Full Scan (TABLE ACCESS FULL)	table: CIUDADES;	1	1.0	cpu_cost = 9191, io_cost = 1

## Consulta segunda: Diagnóstico de la relación del precio, la modalidad de contratación y el tipo de contrato por mes

```
WITH ProcesosMensuales AS (
```

```
SELECT
```

```
EXTRACT(YEAR FROM p.fecha) AS año,
```

```
EXTRACT(MONTH FROM p.fecha) AS mes,
```

```
p.id,
```

```
p.precio_base,
```

```
mc.descripcion AS modalidad,
```

```
tc.descripcion AS tipo_contrato
```

```
FROM Procesos p
```

```
JOIN Modalidades_contratos mc ON mc.id = p.id_modalidad
```

```
JOIN Tipos_contratos tc ON tc.id = p.id_tipo_contrato
```

```
),
```

```
EstadisticasBasicas AS (
```

```
SELECT
```

```
año,
```

```
mes,
```

**COUNT(\*) AS cantidad\_procesos,**  
**MIN(precio\_base) AS precio\_minimo,**  
**ROUND(AVG(precio\_base), 2) AS precio\_promedio,**  
**MAX(precio\_base) AS precio\_maximo**  
**FROM ProcesosMensuales**  
**GROUP BY año, mes**  
**),**

**ModalidadesRecurrentes AS (**  
**SELECT**  
**año,**  
**mes,**  
**modalidad,**  
**COUNT(\*) AS frecuencia,**  
**RANK() OVER (PARTITION BY año, mes ORDER BY COUNT(\*)**  
**DESC) AS ranking\_modalidad**  
**FROM ProcesosMensuales**  
**GROUP BY año, mes, modalidad**  
**),**

**TiposRecurrentes AS (**  
**SELECT**  
**año,**  
**mes,**  
**tipo\_contrato,**  
**COUNT(\*) AS frecuencia,**  
**RANK() OVER (PARTITION BY año, mes ORDER BY COUNT(\*)**  
**DESC) AS ranking\_tipo**  
**FROM ProcesosMensuales**  
**GROUP BY año, mes, tipo\_contrato**

)

***SELECT***

***eb.año,***

***eb.mes,***

***eb.cantidad\_procesos,***

***eb.precio\_minimo,***

***eb.precio\_promedio,***

***eb.precio\_maximo,***

***mr.modalidad AS modalidad\_mas\_recurrente,***

***mr.frecuencia AS frecuencia\_modalidad,***

***tr.tipo\_contrato AS tipo\_contrato\_mas\_recurrente,***

***tr.frecuencia AS frecuencia\_tipo***

***FROM EstadisticasBasicas eb***

***LEFT JOIN ModalidadesRecurrentes mr ON mr.año = eb.año AND mr.mes = eb.mes AND mr.ranking\_modalidad = 1***

***LEFT JOIN TiposRecurrentes tr ON tr.año = eb.año AND tr.mes = eb.mes AND tr.ranking\_tipo = 1***

***ORDER BY eb.año DESC, eb.mes DESC;***

Plan de ejecución:

Resultados: *result\_2.csv*

Posibles mejoras:

Tabla materializada:

-- Primero: Crear tabla de resumen mensual

```
CREATE TABLE resumen_mensual_procesos (
    año NUMBER(4) NOT NULL,
    mes NUMBER(2) NOT NULL,
    cantidad_procesos NUMBER NOT NULL,
    precio_minimo NUMBER(15,2),
    precio_promedio NUMBER(15,2),
    precio_maximo NUMBER(15,2),
    modalidad_mas_recurrente VARCHAR2(50),
    frecuencia_modalidad NUMBER,
    tipo_contrato_mas_recurrente VARCHAR2(50),
    frecuencia_tipo NUMBER,
    fecha_actualizacion DATE DEFAULT SYSDATE,
    CONSTRAINT pk_resumen_mensual PRIMARY KEY (año, mes)
);
```

-- Crear procedimiento para poblar el resumen

CREATE OR REPLACE PROCEDURE actualizar\_resumen\_mensual AS  
BEGIN

-- Eliminar datos existentes para los meses que se actualizarán

DELETE FROM resumen\_mensual\_procesos  
WHERE (año, mes) IN (  
    SELECT EXTRACT(YEAR FROM fecha), EXTRACT(MONTH FROM  
fecha)  
    FROM Procesos  
    WHERE fecha >= ADD\_MONTHS(TRUNC(SYSDATE, 'MM'), -3) --  
Últimos 3 meses  
);

-- Insertar datos actualizados

INSERT INTO resumen\_mensual\_procesos (  
    año, mes, cantidad\_procesos, precio\_minimo, precio\_promedio,  
precio\_maximo,  
    modalidad\_mas\_recurrente, frecuencia\_modalidad,  
tipo\_contrato\_mas\_recurrente, frecuencia\_tipo  
)

WITH ProcesosMensuales AS (

SELECT

EXTRACT(YEAR FROM p.fecha) AS año,

EXTRACT(MONTH FROM p.fecha) AS mes,

p.precio\_base,

mc.descripcion AS modalidad,

tc.descripcion AS tipo\_contrato

FROM Procesos p

JOIN Modalidades\_contratos mc ON mc.id = p.id\_modalidad

JOIN Tipos\_contratos tc ON tc.id = p.id\_tipo\_contrato

*WHERE p.fecha >= ADD\_MONTHS(TRUNC(SYSDATE, 'MM'), -12) --  
Solo último año*

*)*

*SELECT*

*año,*

*mes,*

*COUNT(\*) AS cantidad\_procesos,*

*MIN(precio\_base) AS precio\_minimo,*

*ROUND(AVG(precio\_base), 2) AS precio\_promedio,*

*MAX(precio\_base) AS precio\_maximo,*

*modalidad,*

*frecuencia\_modalidad,*

*tipo\_contrato,*

*frecuencia\_tipo*

*FROM (*

*SELECT*

*año,*

*mes,*

*precio\_base,*

*modalidad,*

*tipo\_contrato,*

*COUNT(\*) OVER (PARTITION BY año, mes) AS total\_mes,*

*ROW\_NUMBER() OVER (PARTITION BY año, mes, modalidad ORDER  
BY COUNT(\*) DESC) AS rn\_modalidad,*

*COUNT(\*) OVER (PARTITION BY año, mes, modalidad) AS  
frecuencia\_modalidad,*

*ROW\_NUMBER() OVER (PARTITION BY año, mes, tipo\_contrato  
ORDER BY COUNT(\*) DESC) AS rn\_tipo,*

*COUNT(\*) OVER (PARTITION BY año, mes, tipo\_contrato) AS  
frecuencia\_tipo*

*FROM ProcesosMensuales*

```

        GROUP BY año, mes, precio_base, modalidad, tipo_contrato
    )

    WHERE rn_modalidad = 1 AND rn_tipo = 1

    GROUP BY año, mes, modalidad, frecuencia_modalidad, tipo_contrato,
    frecuencia_tipo;

    COMMIT;

END actualizar_resumen_mensual;

/

-- Crear job programado para actualizar nightly
BEGIN

    DBMS_SCHEDULER.CREATE_JOB (

        job_name      => 'ACTUALIZAR_RESUMEN_MENSUAL_JOB',
        job_type       => 'STORED_PROCEDURE',
        job_action      => 'actualizar_resumen_mensual',
        start_date      => SYSTIMESTAMP,
        repeat_interval => 'FREQ=DAILY;BYHOUR=2;BYMINUTE=0',
        enabled         => TRUE

    );

END;

/

-- Consulta FINAL ultra-rápida
SELECT

    año,

    mes,

    cantidad_procesos,

    precio_minimo,

```



*precio\_promedio,*

*precio\_maximo,*

*modalidad\_mas\_recurrente,*

*frecuencia\_modalidad,*

*tipo\_contrato\_mas\_recurrente,*

*frecuencia\_tipo*

*FROM resumen\_mensual\_procesos*

*ORDER BY año DESC, mes DESC;*

Operation	Params	Rows	Total Cost	Raw Desc
↵ Select		1	3.0	cpu_cost = 42485398, io_cost = 2
√ Order By (SORT ORDER BY)		1	3.0	cpu_cost = 42485398, io_cost = 2
Full Scan (TABLE ACCESS FULL)	table: RESUMEN_MENSUAL_PROCESOS;	1	2.0	cpu_cost = 7121, io_cost = 2