LAS VEGAS

regoUniversity 2025

Sponsored by ValueOps by Broadcom | Clarity by Broadcom | Rally by Broadcom | ConnectALL by Broadcom | Insights by Broadcom

SQL Advanced Topics

Your Guides:
Shekhar Kanade & David Matzdorf

Gold Sponsor aws

Introductions

- Take 5 Minutes
- Turn to a Person Near You
- Introduce Yourself

Agenda

- Introduction
- Inline Views
- Common Table Expressions (CTEs)
- Analytic Functions
- Hierarchical Queries

Introduction

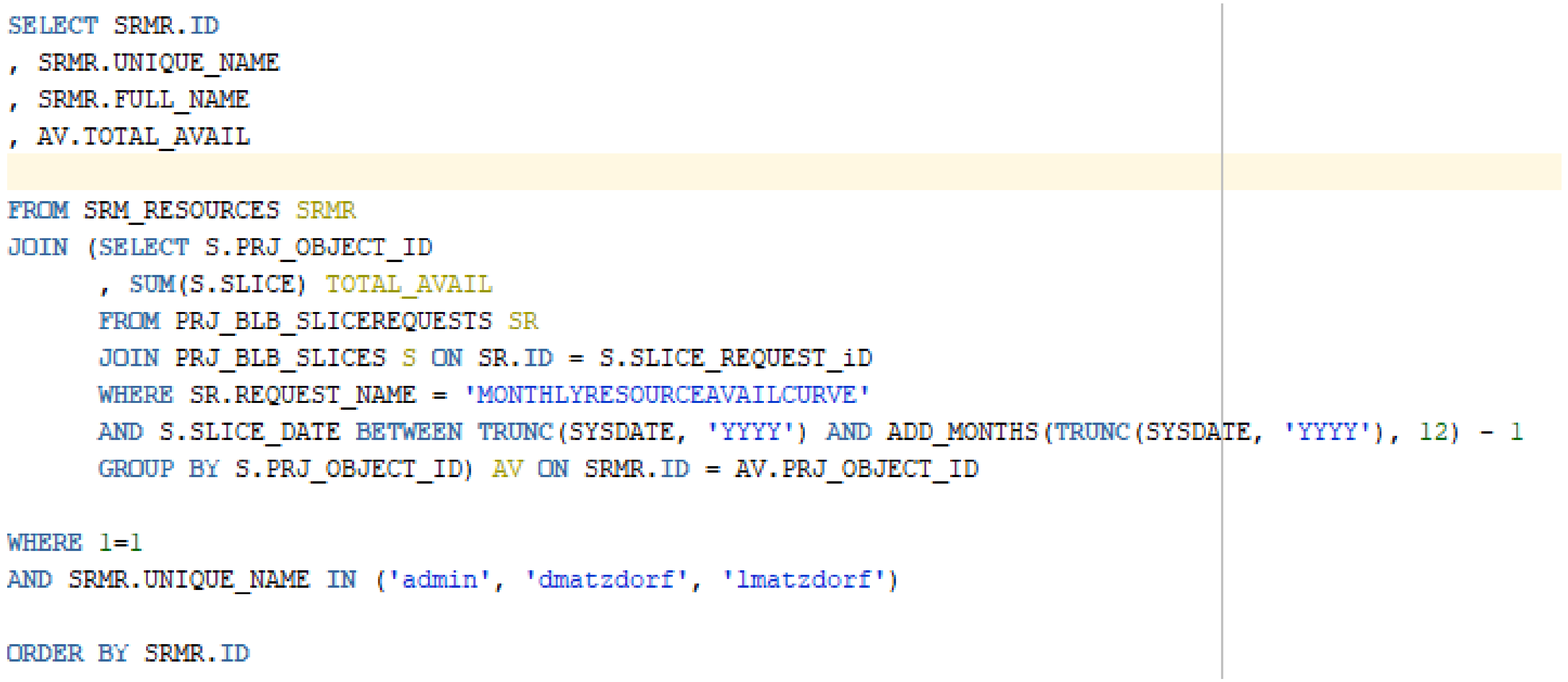
LAS VEGAS
regoUniversity
2025

Inline Views

What is an Inline View

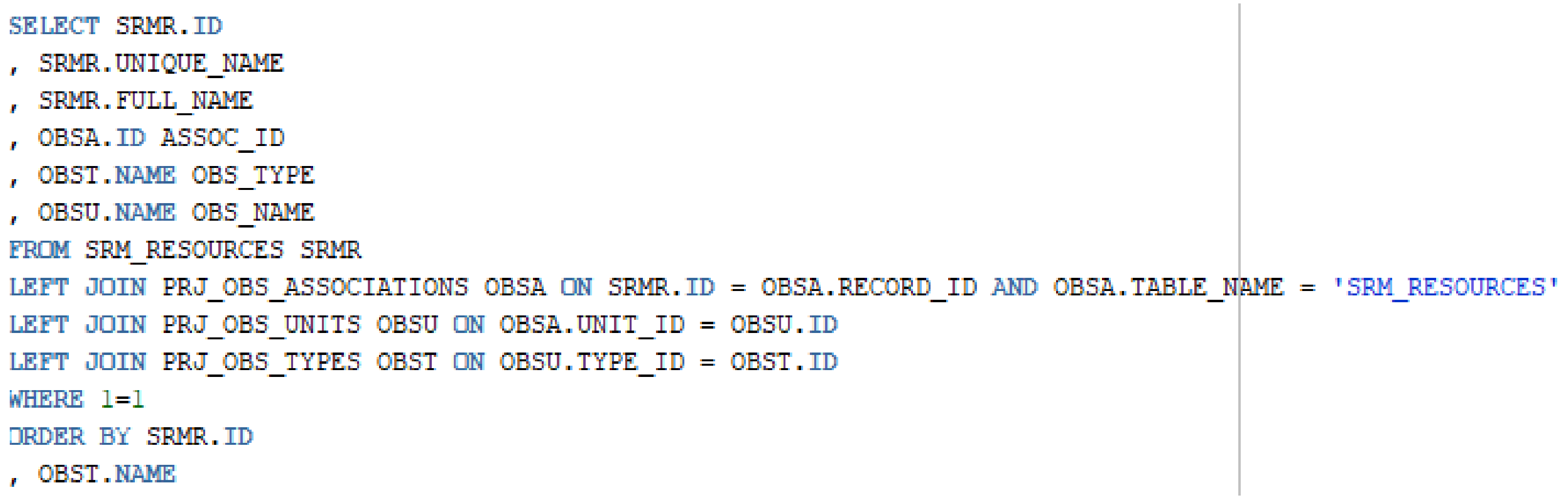
- Subquery written in the FROM clause
- Acts like a temp table that exists only while the query runs
- Not stored within the database like a traditional view
  - Pre-aggregating data
  - Filtering before joining
  - Avoiding Repetitive Subqueries

Pre-aggregating data

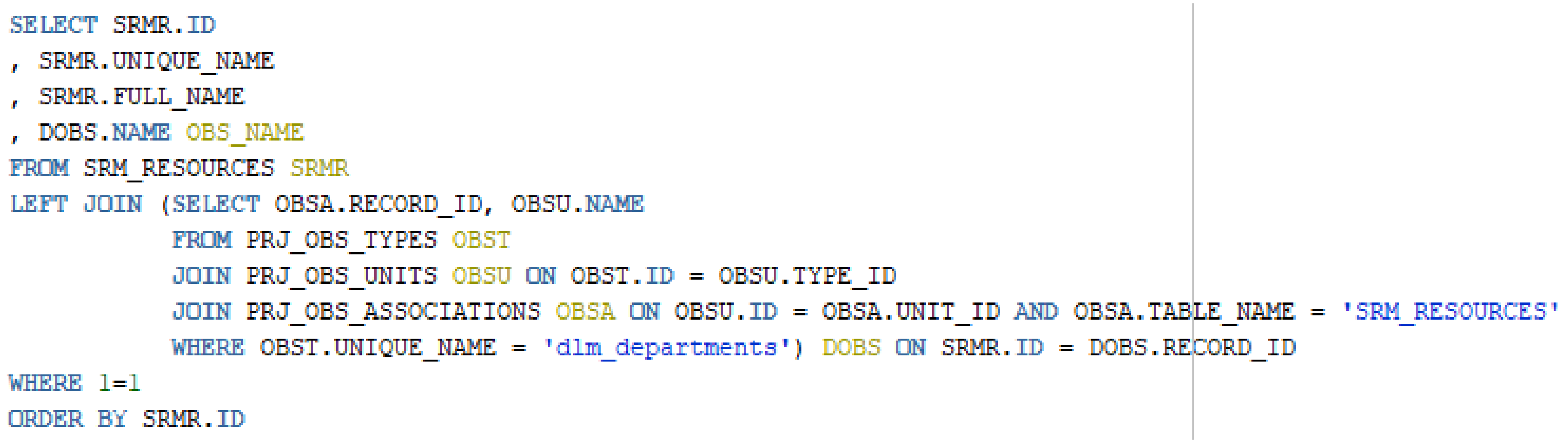


```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, AV.TOTAL_AVAIL

FROM SRM_RESOURCES SRMR
JOIN (SELECT S.PRJ_OBJECT_ID
      , SUM(S.SLICE) TOTAL_AVAIL
      FROM PRJ_BLB_SLICEREQUESTS SR
      JOIN PRJ_BLB_SLICES S ON SR.ID = S.SLICE_REQUEST_iD
      WHERE SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
      AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
      GROUP BY S.PRJ_OBJECT_ID) AV ON SRMR.ID = AV.PRJ_OBJECT_ID

WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')

ORDER BY SRMR.ID
```

Filtering Before Joining



```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, OBSA.ID ASSOC_ID
, OBST.NAME OBS_TYPE
, OBSU.NAME OBS_NAME
FROM SRM_RESOURCES SRMR
LEFT JOIN PRJ_OBS_ASSOCIATIONS OBSA ON SRMR.ID = OBSA.RECORD_ID AND OBSA.TABLE_NAME = 'SRM_RESOURCES'
LEFT JOIN PRJ_OBS_UNITS OBSU ON OBSA.UNIT_ID = OBSU.ID
LEFT JOIN PRJ_OBS_TYPES OBST ON OBSU.TYPE_ID = OBST.ID
WHERE 1=1
ORDER BY SRMR.ID
, OBST.NAME
```



```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, DOBS.NAME OBS_NAME
FROM SRM_RESOURCES SRMR
LEFT JOIN (SELECT OBSA.RECORD_ID, OBSU.NAME
           FROM PRJ_OBS_TYPES OBST
           JOIN PRJ_OBS_UNITS OBSU ON OBST.ID = OBSU.TYPE_ID
           JOIN PRJ_OBS_ASSOCIATIONS OBSA ON OBSU.ID = OBSA.UNIT_ID AND OBSA.TABLE_NAME = 'SRM_RESOURCES'
           WHERE OBST.UNIQUE_NAME = 'dlm_departments') DOBS ON SRMR.ID = DOBS.RECORD_ID
WHERE 1=1
ORDER BY SRMR.ID
```

Avoiding Repetitive Subqueries

Without Inline View

```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, (SELECT SUM(A.PRESTSUM)
   FROM PRASSIGNMENT A
   WHERE A.PRRESOURCEID = SRMR.ID) / 3600 TOTAL_ETCS
, (SELECT SUM(A.PRACTSUM)
   FROM PRASSIGNMENT A
   WHERE A.PRRESOURCEID = SRMR.ID) / 3600 TOTAL_ACT
FROM SRM_RESOURCES SRMR
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
ORDER BY SRMR.ID
```

With Inline View

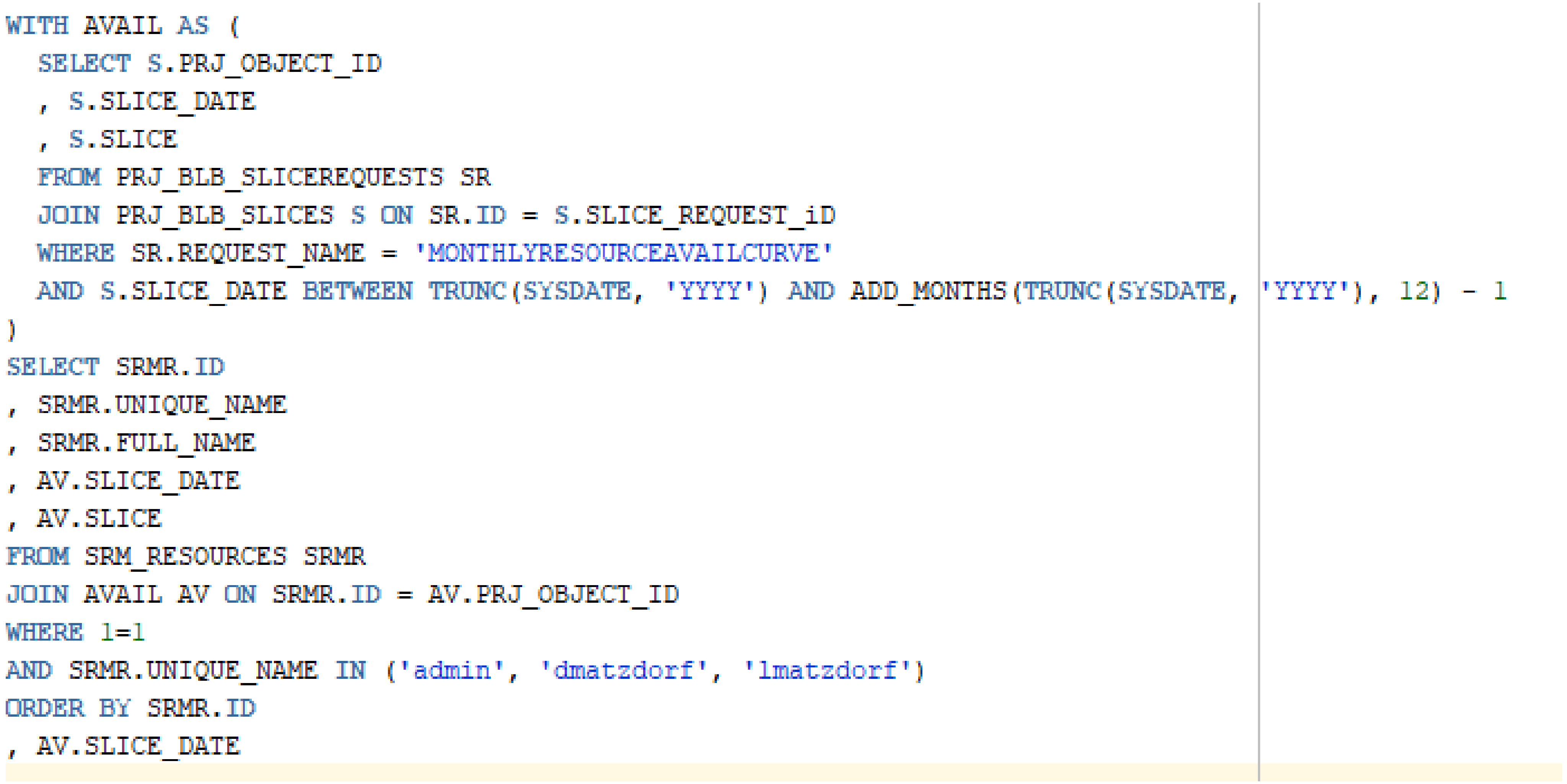
```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, A.TOTAL_ETCS
, A.TOTAL_ACT
FROM SRM_RESOURCES SRMR
LEFT JOIN (SELECT A.PRRESOURCEID
           , SUM(A.PRESTSUM) / 3600 TOTAL_ETCS
           , SUM(A.PRACTSUM) / 3600 TOTAL_ACT
           FROM PRASSIGNMENT A
           WHERE 1=1
           GROUP BY A.PRRESOURCEID) A ON SRMR.ID = A.PRRESOURCEID
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
ORDER BY SRMR.ID
```

Common Table Expressions (CTEs)

What is a CTE

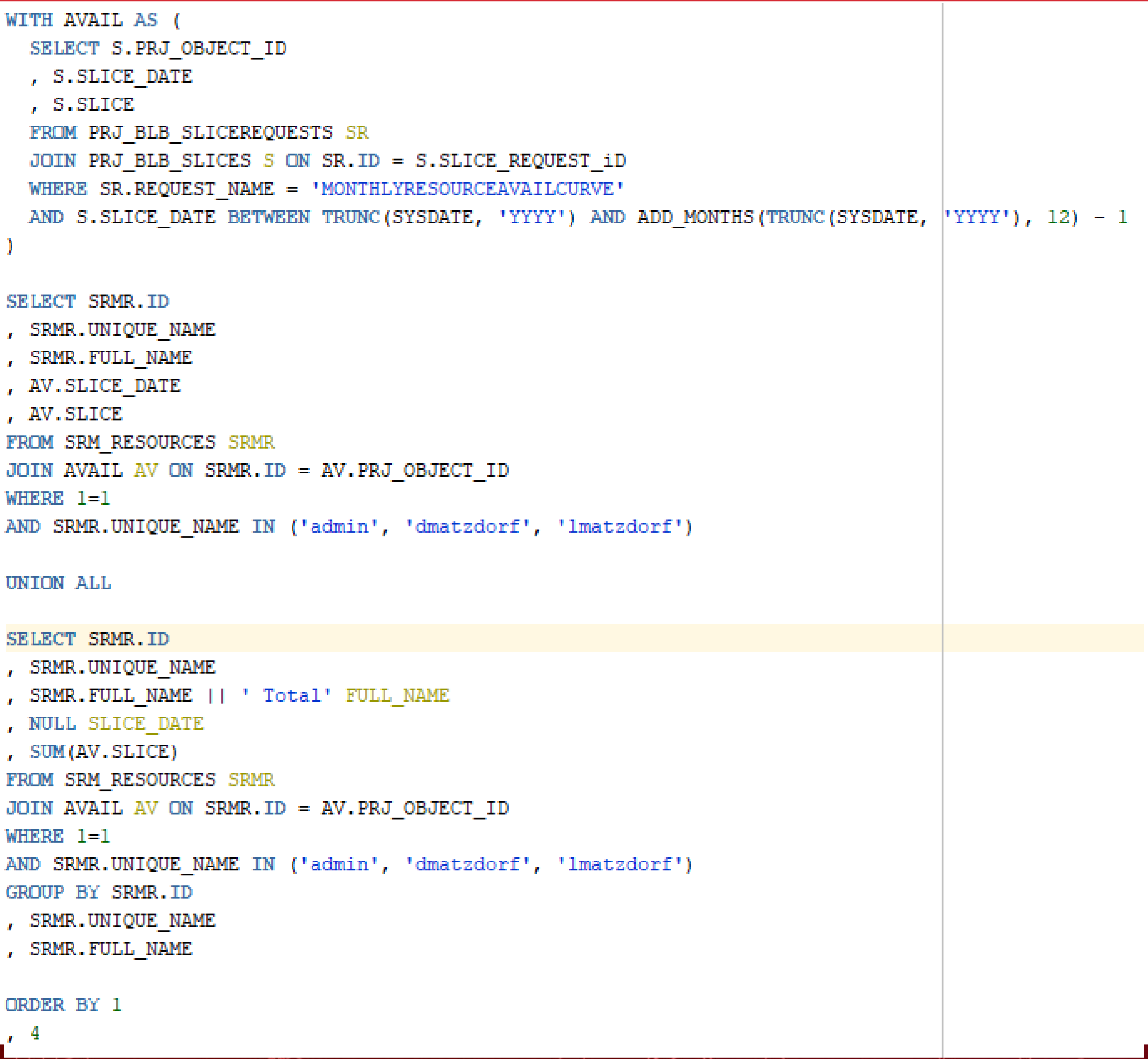
- Named temporary result (temporary view)
- Only exist for the duration of the query in which they are defined
- Begins with the keyword WITH
- Can be referenced multiple times with the same query
- Advantages
  - Makes complex queries easier to read and organize
  - Allows you to break down queries into logical steps
  - Aids in performance for queries that access the same data multiple times

CTE Example



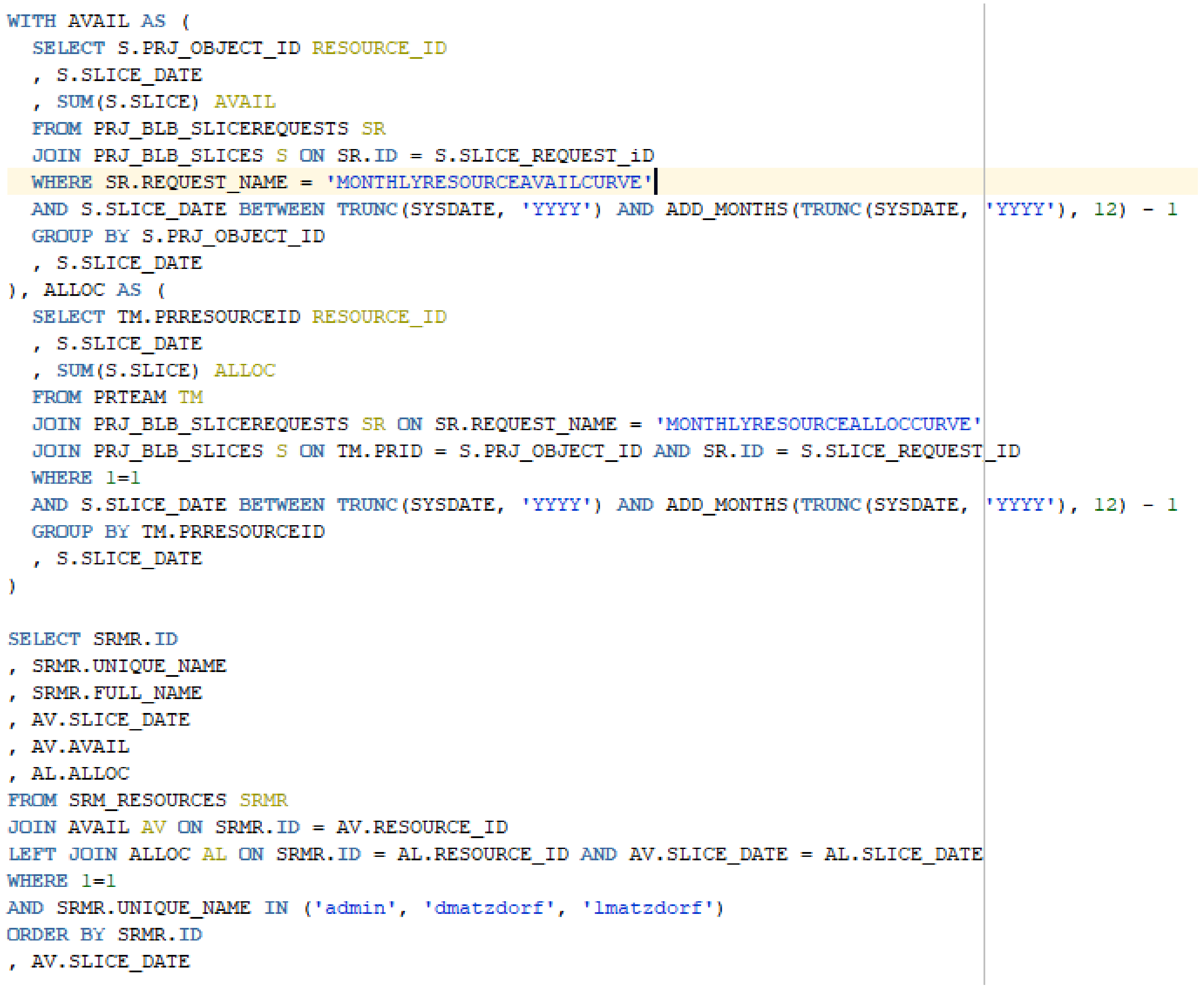
```
WITH AVAIL AS (
  SELECT S.PRJ_OBJECT_ID
  , S.SLICE_DATE
  , S.SLICE
  FROM PRJ_BLB_SLICEREQUESTS SR
  JOIN PRJ_BLB_SLICES S ON SR.ID = S.SLICE_REQUEST_iD
  WHERE SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
  AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
)
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, AV.SLICE_DATE
, AV.SLICE
FROM SRM_RESOURCES SRMR
JOIN AVAIL AV ON SRMR.ID = AV.PRJ_OBJECT_ID
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
ORDER BY SRMR.ID
, AV.SLICE_DATE
```

CTE - Reuse



```
WITH AVAIL AS (
  SELECT S.PRJ_OBJECT_ID
  , S.SLICE_DATE
  , S.SLICE
  FROM PRJ_BLB_SLICEREQUESTS SR
  JOIN PRJ_BLB_SLICES S ON SR.ID = S.SLICE_REQUEST_iD
  WHERE SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
  AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
)

SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, AV.SLICE_DATE
, AV.SLICE
FROM SRM_RESOURCES SRMR
JOIN AVAIL AV ON SRMR.ID = AV.PRJ_OBJECT_ID
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')

UNION ALL

SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME || ' Total' FULL_NAME
, NULL SLICE_DATE
, SUM(AV.SLICE)
FROM SRM_RESOURCES SRMR
JOIN AVAIL AV ON SRMR.ID = AV.PRJ_OBJECT_ID
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
GROUP BY SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME

ORDER BY 1
, 4
```

Multiple CTEs



```sql
WITH AVAIL AS (
  SELECT S.PRJ_OBJECT_ID RESOURCE_ID
  , S.SLICE_DATE
  , SUM(S.SLICE) AVAIL
  FROM PRJ_BLB_SLICEREQUESTS SR
  JOIN PRJ_BLB_SLICES S ON SR.ID = S.SLICE_REQUEST_iD
  WHERE SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
  AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
  GROUP BY S.PRJ_OBJECT_ID
  , S.SLICE_DATE
), ALLOC AS (
  SELECT TM.PRRESOURCEID RESOURCE_ID
  , S.SLICE_DATE
  , SUM(S.SLICE) ALLOC
  FROM PRTEAM TM
  JOIN PRJ_BLB_SLICEREQUESTS SR ON SR.REQUEST_NAME = 'MONTHLYRESOURCEALLOCCURVE'
  JOIN PRJ_BLB_SLICES S ON TM.PRID = S.PRJ_OBJECT_ID AND SR.ID = S.SLICE_REQUEST_ID
  WHERE 1=1
  AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
  GROUP BY TM.PRRESOURCEID
  , S.SLICE_DATE
)

SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, AV.SLICE_DATE
, AV.AVAIL
, AL.ALLOC
FROM SRM_RESOURCES SRMR
JOIN AVAIL AV ON SRMR.ID = AV.RESOURCE_ID
LEFT JOIN ALLOC AL ON SRMR.ID = AL.RESOURCE_ID AND AV.SLICE_DATE = AL.SLICE_DATE
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
ORDER BY SRMR.ID
, AV.SLICE_DATE
```

Analytic Functions

What Are Analytic Functions

- SQL functions that perform calculations across a set of rows that are related to the current row
- Unlike aggregate functions they return a value for each row while still considering other rows in the calculation
- In other words, they do not collapse rows into a single result
- Often involve ranking, cumulative, or moving calculations
- Allow advanced calculations without losing row-level detail
  - i.e.: ranking, running totals, moving averages, comparisons to other rows
- Can only appear in the SELECT or ORDER BY clause

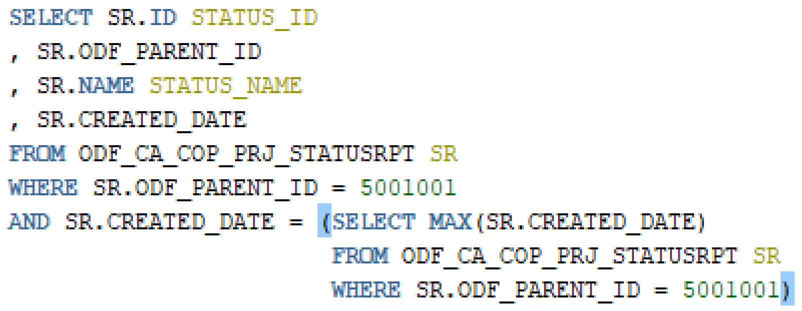
Available Functions

- AVG
- CORR
- COUNT
- COVAR_POP
- COVAR_SAMP
- CUME_DIST
- DENSE_RANK
- FIRST
- FIRST_VALUE
- LAG
- LAST
- LAST_VALUE
- LEAD
- LISTAGG
- MAX
- MEDIAN
- MIN
- NTH_VALUE
- NTILE
- PERCENT_RANK
- PERCENTILE_CONT
- PERCENTILE_DISC
- RANK
- RATIO_TO_REPORT
- REGR_
- ROW_NUMBER
- STDDEV
- STDDEV_POP
- STDDEV_SAMP
- SUM
- VAR_POP
- VAR_SAMP
- VARIANCE

Why Use Analytic Functions

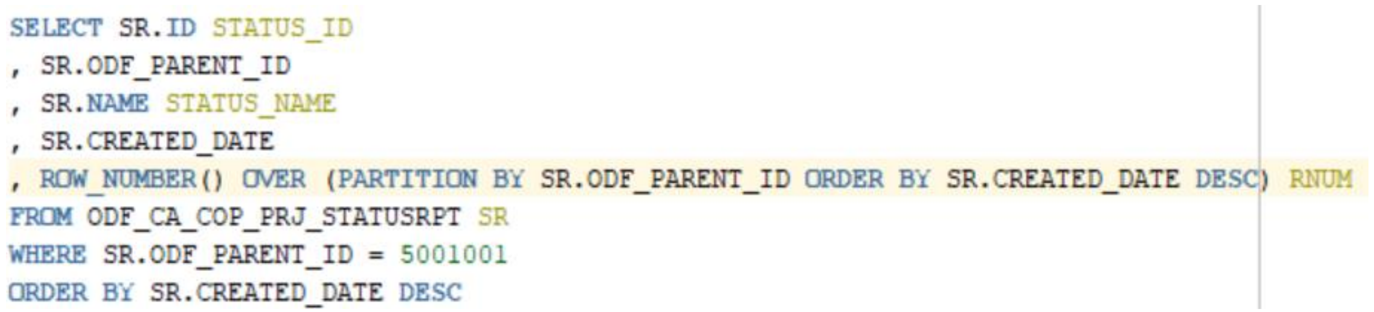
- Can be done with native SQL
- Odd syntax
- Analytic functions are faster and more accurate
- Get the latest status report
  - Get the max updated date for each project and join to it
  - Not accurate if there are multiple reports updated at the same time
  - Not efficient

Traditional Approach



```sql
SELECT SR.ID STATUS_ID
, SR.ODF_PARENT_ID
, SR.NAME STATUS_NAME
, SR.CREATED_DATE
FROM ODF_CA_COP_PRJ_STATUSRPT SR
WHERE SR.ODF_PARENT_ID = 5001001
AND SR.CREATED_DATE = (SELECT MAX(SR.CREATED_DATE)
                       FROM ODF_CA_COP_PRJ_STATUSRPT SR
                       WHERE SR.ODF_PARENT_ID = 5001001)
```

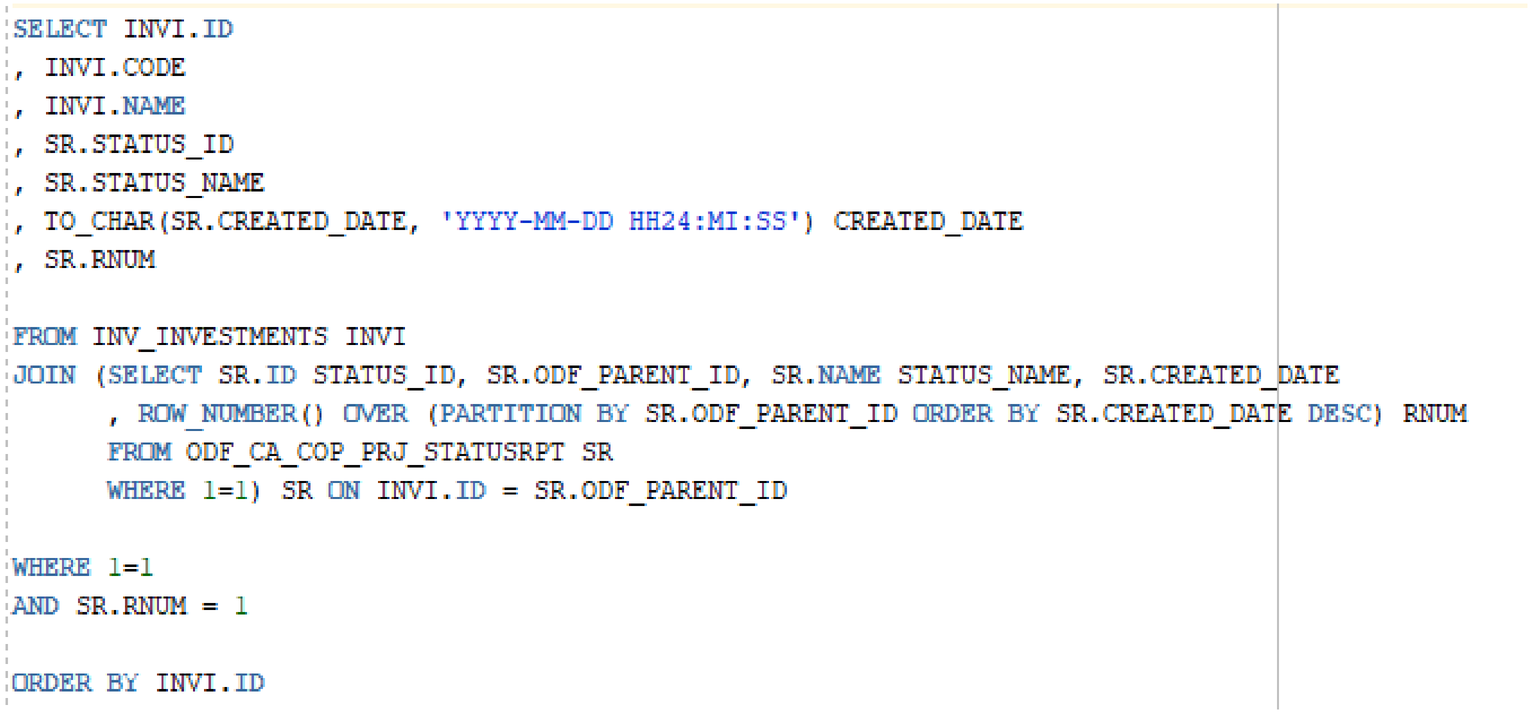
Analytic Function Approach



```sql
SELECT SR.ID STATUS_ID
, SR.ODF_PARENT_ID
, SR.NAME STATUS_NAME
, SR.CREATED_DATE
, ROW_NUMBER() OVER (PARTITION BY SR.ODF_PARENT_ID ORDER BY SR.CREATED_DATE DESC) RNUM
FROM ODF_CA_COP_PRJ_STATUSRPT SR
WHERE SR.ODF_PARENT_ID = 5001001
ORDER BY SR.CREATED_DATE DESC
```

Latest Status Report

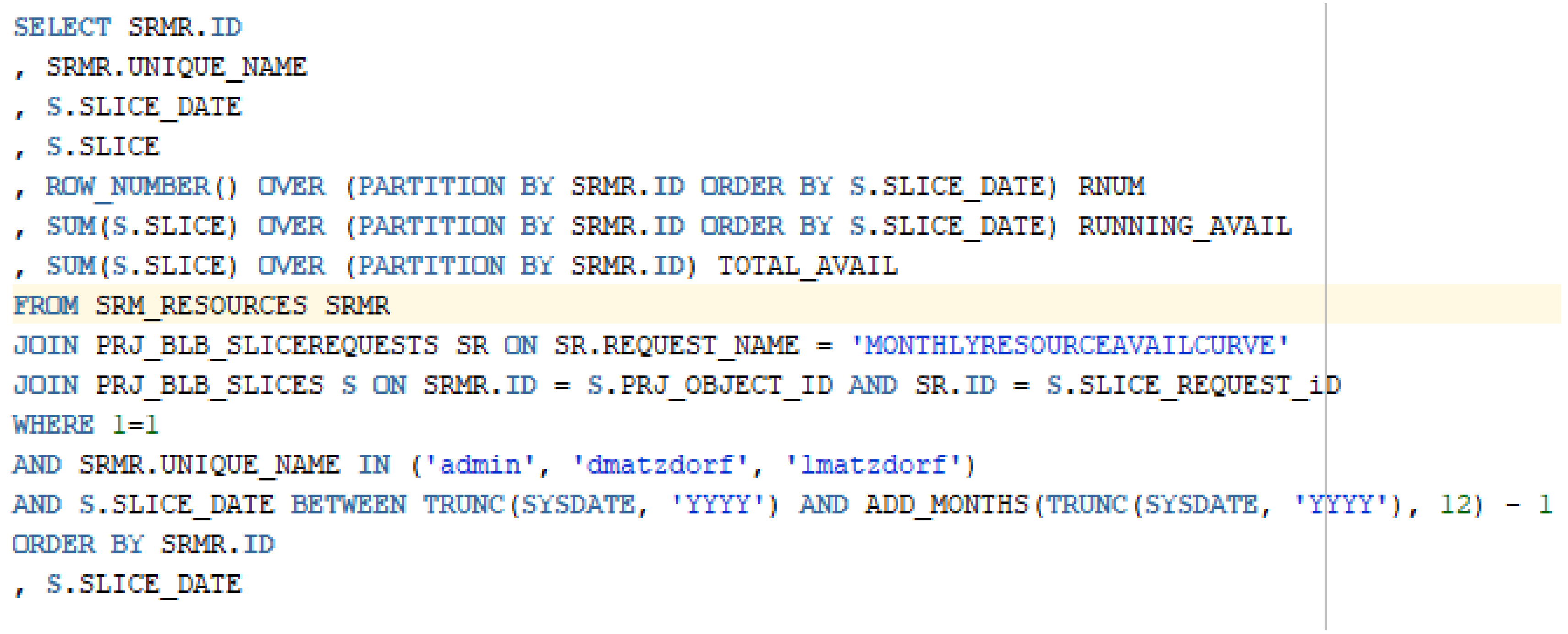
- Most recent status report by project



```
SELECT INVI.ID
, INVI.CODE
, INVI.NAME
, SR.STATUS_ID
, SR.STATUS_NAME
, TO_CHAR(SR.CREATED_DATE, 'YYYY-MM-DD HH24:MI:SS') CREATED_DATE
, SR.RNUM

FROM INV_INVESTMENTS INVI
JOIN (SELECT SR.ID STATUS_ID, SR.ODF_PARENT_ID, SR.NAME STATUS_NAME, SR.CREATED_DATE
      , ROW_NUMBER() OVER (PARTITION BY SR.ODF_PARENT_ID ORDER BY SR.CREATED_DATE DESC) RNUM
      FROM ODF_CA_COP_PRJ_STATUSRPT SR
      WHERE 1=1) SR ON INVI.ID = SR.ODF_PARENT_ID

WHERE 1=1
AND SR.RNUM = 1

ORDER BY INVI.ID
```

Running Totals

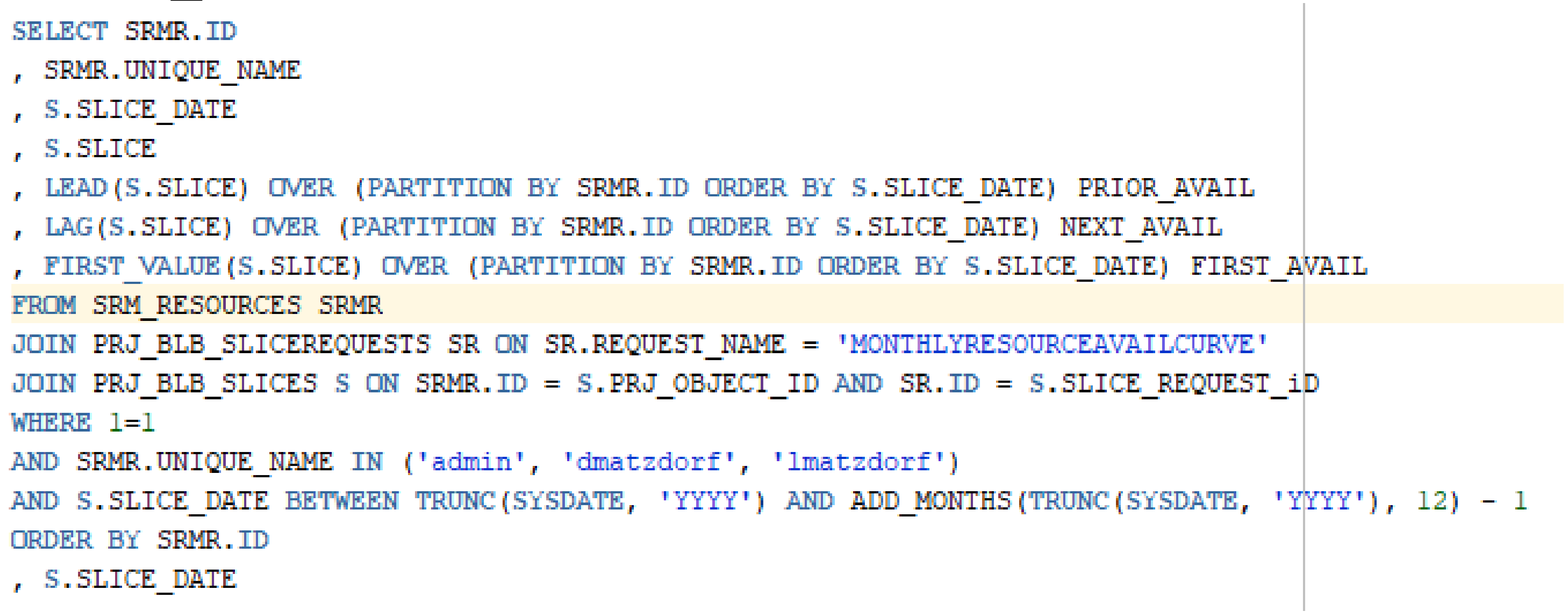
- ROW_NUMBER()
- Running Availability
- Total Availability By Resource



```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, S.SLICE_DATE
, S.SLICE
, ROW_NUMBER() OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE) RNUM
, SUM(S.SLICE) OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE) RUNNING_AVAIL
, SUM(S.SLICE) OVER (PARTITION BY SRMR.ID) TOTAL_AVAIL
FROM SRM_RESOURCES SRMR
JOIN PRJ_BLB_SLICEREQUESTS SR ON SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
JOIN PRJ_BLB_SLICES S ON SRMR.ID = S.PRJ_OBJECT_ID AND SR.ID = S.SLICE_REQUEST_ID
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
ORDER BY SRMR.ID
, S.SLICE_DATE
```

LEAD / LAG

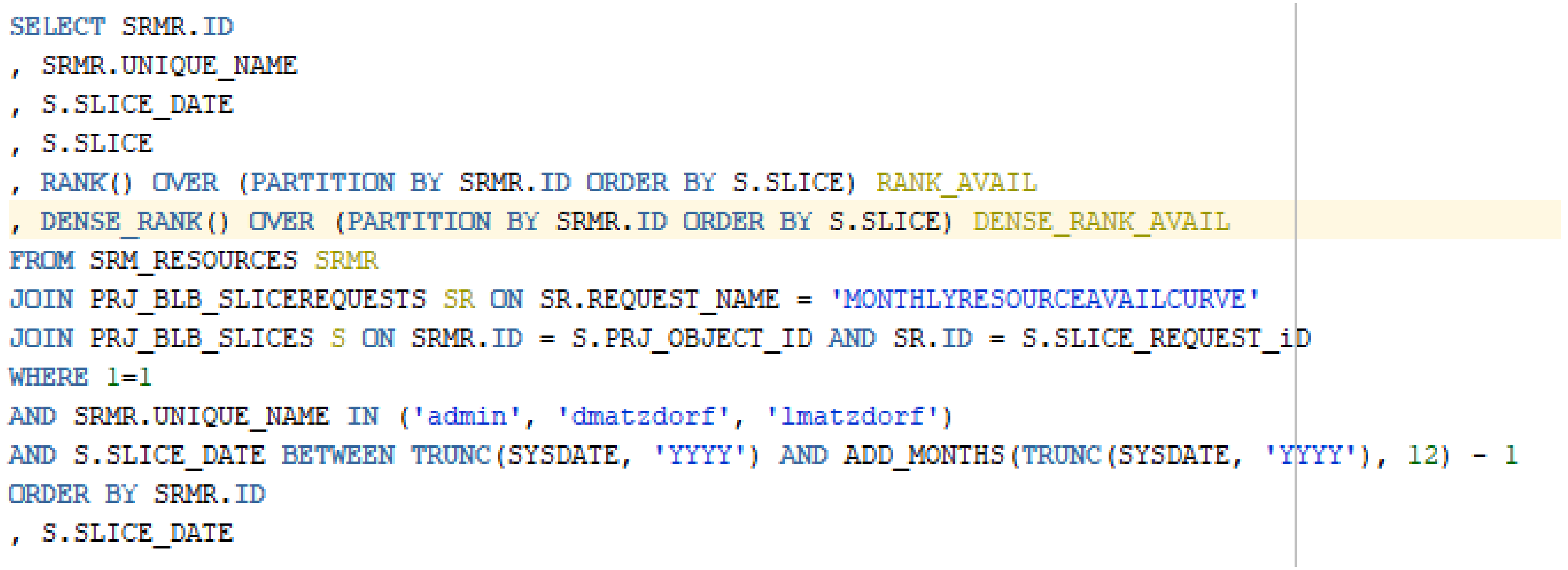
- LEAD
- LAG
- FIRST_VALUE



```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, S.SLICE_DATE
, S.SLICE
, LEAD(S.SLICE) OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE) PRIOR_AVAIL
, LAG(S.SLICE) OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE) NEXT_AVAIL
, FIRST_VALUE(S.SLICE) OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE) FIRST_AVAIL
FROM SRM_RESOURCES SRMR
JOIN PRJ_BLB_SLICEREQUESTS SR ON SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
JOIN PRJ_BLB_SLICES S ON SRMR.ID = S.PRJ_OBJECT_ID AND SR.ID = S.SLICE_REQUEST_iD
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
ORDER BY SRMR.ID
, S.SLICE_DATE
```

Ranking

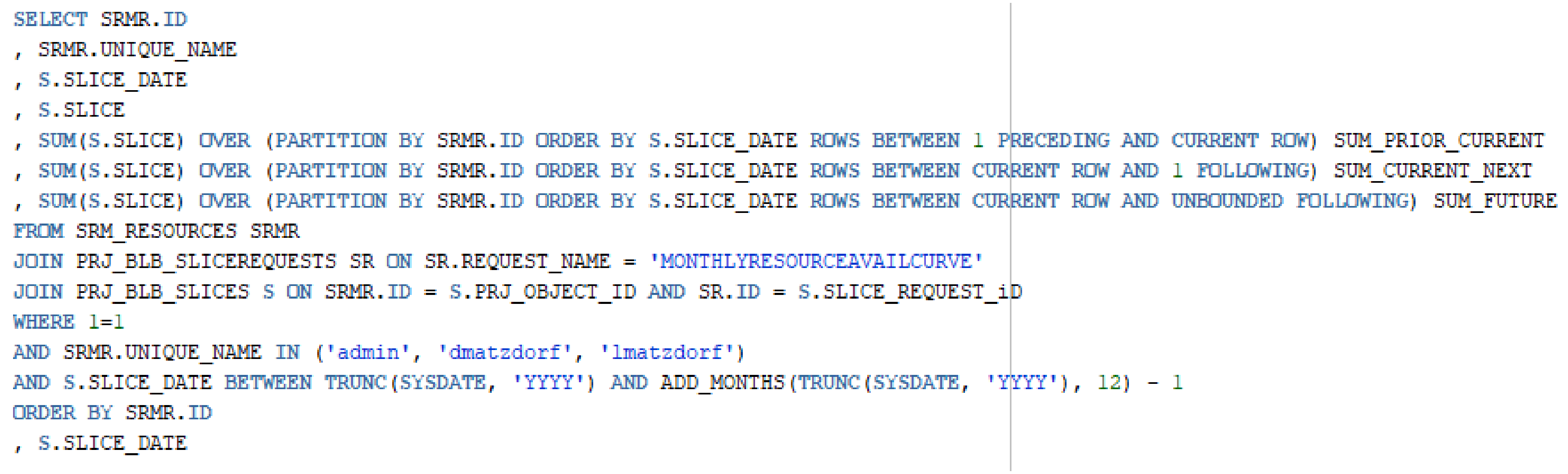
- RANK
  - Leaves gaps in the ranking sequence when ties occur
- DENSE_RANK
  - Does not leave gaps in the ranking sequence when ties occur



```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, S.SLICE_DATE
, S.SLICE
, RANK() OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE) RANK_AVAIL
, DENSE_RANK() OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE) DENSE_RANK_AVAIL
FROM SRM_RESOURCES SRMR
JOIN PRJ_BLB_SLICEREQUESTS SR ON SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
JOIN PRJ_BLB_SLICES S ON SRMR.ID = S.PRJ_OBJECT_ID AND SR.ID = S.SLICE_REQUEST_iD
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
ORDER BY SRMR.ID
, S.SLICE_DATE
```

Window Functions

- Operate on a defined set of rows
  - Current Row
  - Unbounded Preceding
  - Offset Preceding
  - Unbounded Following
  - Offset Following



```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, S.SLICE_DATE
, S.SLICE
, SUM(S.SLICE) OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) SUM_PRIOR_CURRENT
, SUM(S.SLICE) OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE ROWS BETWEEN CURRENT ROW AND 1 FOLLOWING) SUM_CURRENT_NEXT
, SUM(S.SLICE) OVER (PARTITION BY SRMR.ID ORDER BY S.SLICE_DATE ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) SUM_FUTURE
FROM SRM_RESOURCES SRMR
JOIN PRJ_BLB_SLICEREQUESTS SR ON SR.REQUEST_NAME = 'MONTHLYRESOURCEAVAILCURVE'
JOIN PRJ_BLB_SLICES S ON SRMR.ID = S.PRJ_OBJECT_ID AND SR.ID = S.SLICE_REQUEST_iD
WHERE 1=1
AND SRMR.UNIQUE_NAME IN ('admin', 'dmatzdorf', 'lmatzdorf')
AND S.SLICE_DATE BETWEEN TRUNC(SYSDATE, 'YYYY') AND ADD_MONTHS(TRUNC(SYSDATE, 'YYYY'), 12) - 1
ORDER BY SRMR.ID
, S.SLICE_DATE
```

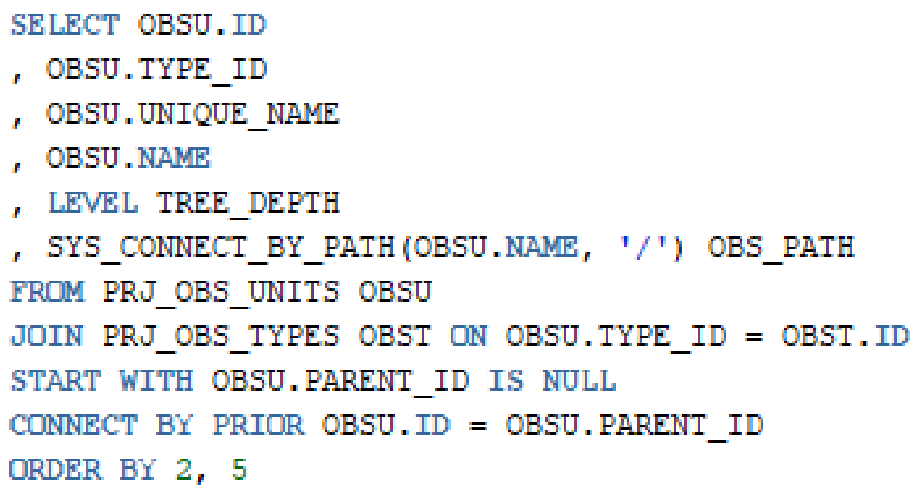
Hierarchical Queries

Hierarchical Queries

- Used to receive data with a parent-child relationship
- Allow results to show levels of data
- Useful for showing
  - OBS Data
  - Calculating the OBS Path
  - Showing reporting structure

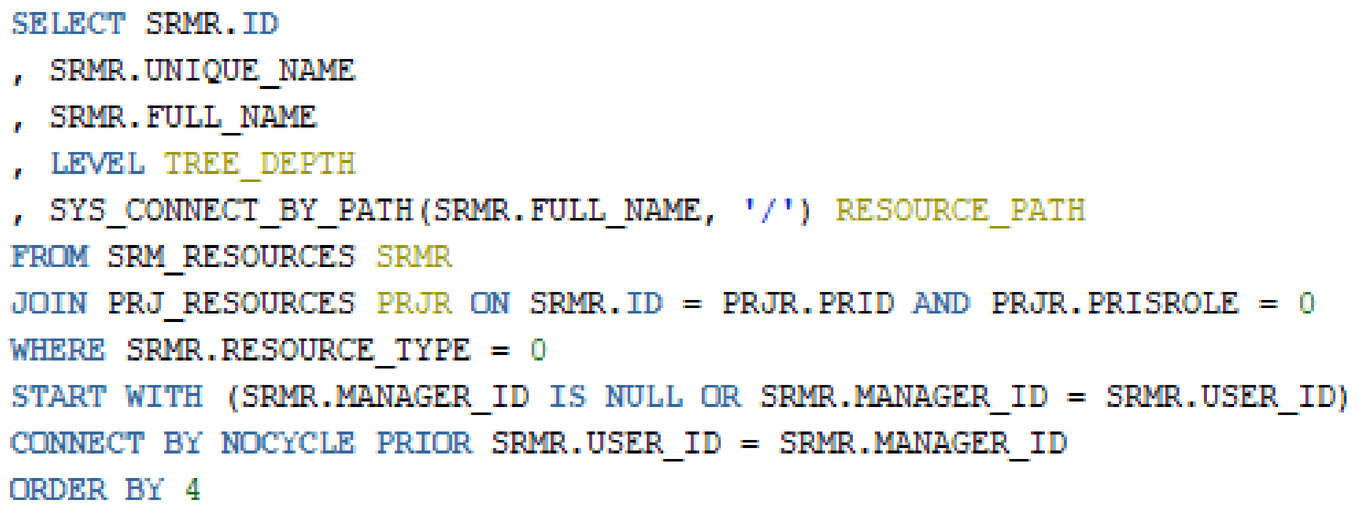
Oracle CONNECT BY

OBS Path



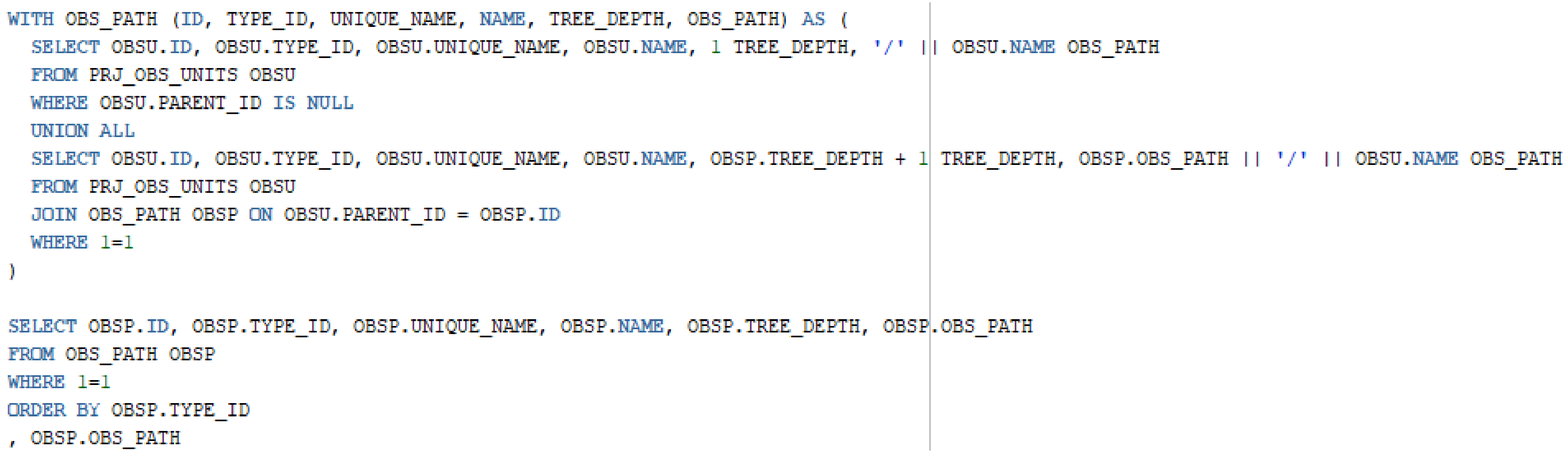
```
SELECT OBSU.ID
, OBSU.TYPE_ID
, OBSU.UNIQUE_NAME
, OBSU.NAME
, LEVEL TREE_DEPTH
, SYS_CONNECT_BY_PATH(OBSU.NAME, '/') OBS_PATH
FROM PRJ_OBS_UNITS OBSU
JOIN PRJ_OBS_TYPES OBST ON OBSU.TYPE_ID = OBST.ID
START WITH OBSU.PARENT_ID IS NULL
CONNECT BY PRIOR OBSU.ID = OBSU.PARENT_ID
ORDER BY 2, 5
```

Resource Reporting Structure



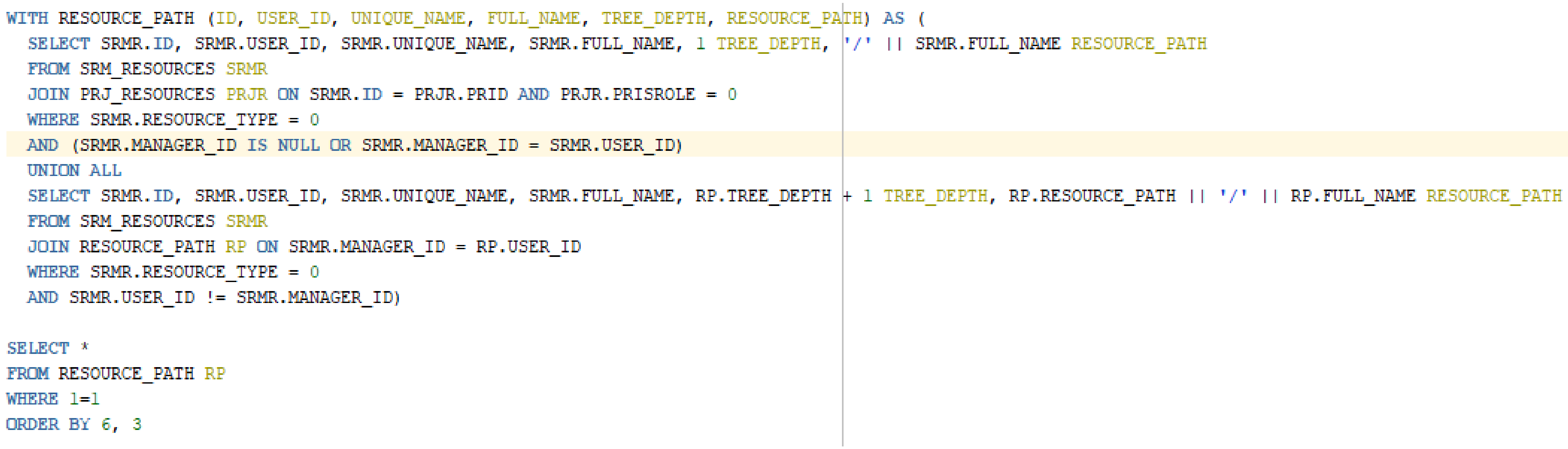
```
SELECT SRMR.ID
, SRMR.UNIQUE_NAME
, SRMR.FULL_NAME
, LEVEL TREE_DEPTH
, SYS_CONNECT_BY_PATH(SRMR.FULL_NAME, '/') RESOURCE_PATH
FROM SRM_RESOURCES SRMR
JOIN PRJ_RESOURCES PRJR ON SRMR.ID = PRJR.PRID AND PRJR.PRISROLE = 0
WHERE SRMR.RESOURCE_TYPE = 0
START WITH (SRMR.MANAGER_ID IS NULL OR SRMR.MANAGER_ID = SRMR.USER_ID)
CONNECT BY NOCYCLE PRIOR SRMR.USER_ID = SRMR.MANAGER_ID
ORDER BY 4
```

Recursive CTE – OBS Path



```
WITH OBS_PATH (ID, TYPE_ID, UNIQUE_NAME, NAME, TREE_DEPTH, OBS_PATH) AS (
  SELECT OBSU.ID, OBSU.TYPE_ID, OBSU.UNIQUE_NAME, OBSU.NAME, 1 TREE_DEPTH, '/' || OBSU.NAME OBS_PATH
  FROM PRJ_OBS_UNITS OBSU
  WHERE OBSU.PARENT_ID IS NULL
  UNION ALL
  SELECT OBSU.ID, OBSU.TYPE_ID, OBSU.UNIQUE_NAME, OBSU.NAME, OBSP.TREE_DEPTH + 1 TREE_DEPTH, OBSP.OBS_PATH || '/' || OBSU.NAME OBS_PATH
  FROM PRJ_OBS_UNITS OBSU
  JOIN OBS_PATH OBSP ON OBSU.PARENT_ID = OBSP.ID
  WHERE 1=1
)

SELECT OBSP.ID, OBSP.TYPE_ID, OBSP.UNIQUE_NAME, OBSP.NAME, OBSP.TREE_DEPTH, OBSP.OBS_PATH
FROM OBS_PATH OBSP
WHERE 1=1
ORDER BY OBSP.TYPE_ID
, OBSP.OBS_PATH
```

Recursive CTE – Resource Reporting Structure



```sql
WITH RESOURCE_PATH (ID, USER_ID, UNIQUE_NAME, FULL_NAME, TREE_DEPTH, RESOURCE_PATH) AS (
  SELECT SRMR.ID, SRMR.USER_ID, SRMR.UNIQUE_NAME, SRMR.FULL_NAME, 1 TREE_DEPTH, '/' || SRMR.FULL_NAME RESOURCE_PATH
  FROM SRM_RESOURCES SRMR
  JOIN PRJ_RESOURCES PRJR ON SRMR.ID = PRJR.PRID AND PRJR.PRISROLE = 0
  WHERE SRMR.RESOURCE_TYPE = 0
  AND (SRMR.MANAGER_ID IS NULL OR SRMR.MANAGER_ID = SRMR.USER_ID)
  UNION ALL
  SELECT SRMR.ID, SRMR.USER_ID, SRMR.UNIQUE_NAME, SRMR.FULL_NAME, RP.TREE_DEPTH + 1 TREE_DEPTH, RP.RESOURCE_PATH || '/' || RP.FULL_NAME RESOURCE_PATH
  FROM SRM_RESOURCES SRMR
  JOIN RESOURCE_PATH RP ON SRMR.MANAGER_ID = RP.USER_ID
  WHERE SRMR.RESOURCE_TYPE = 0
  AND SRMR.USER_ID != SRMR.MANAGER_ID)

SELECT *
FROM RESOURCE_PATH RP
WHERE 1=1
ORDER BY 6, 3
```

regoUniversity

Master Clarity with Rego University

Earn Certifications in Administration, Leadership, and Technical Proficiency

Let Rego be your guide.

Elevate Your Professional Expertise with Rego University Certifications

Rego is excited to continue our **certification programs**, designed to enhance your expertise in Clarity administration, leadership, and technical skills. These certifications provide hands-on experience and knowledge to excel in your career.

regoUniversity
Best Practice
Clarity Administrator
Certification
Program

regoUniversity
Best Practice
Clarity Lead
Certification
Program

regoUniversity
Best Practice
Clarity Technical
Certification
Program

Certification Requirements:

- ✓ **Completion**: 12 units per certification track
- ✓ **Eligibility**: Open to all Rego University attendees

Important Reminder:

To have your certification **credits tracked**, ensure you **complete the class surveys in the app** after each session. This step is critical for certification progress.

Questions?

LAS VEGAS regoUniversity 2025

Surveys

Please take a few moments to fill out the class survey.
Your feedback is extremely important for future events.

Thank You For Attending Rego University

Instructions for PMI credits

- Access your account at pmi.org
- Click on **Certifications**
- Click on **Maintain My Certification**
- Click on **Visit CCR's** button under the **Report PDU's**
- Click on **Report PDU's**
- Click on **Course or Training**
- Class Provider = **Rego Consulting**
- Class Name = **Rego University**
- Course **Description**
- Date Started = **Today's Date**
- Date Completed = **Today's Date**
- Hours Completed = **1 PDU per hour of class time**
- Training classes = **Technical**
- Click on **I agree** and **Submit**

Let us know how we can improve!
Don't forget to fill out the class survey.

Phone
888.813.0444

Email
info@regoconsulting.com

Website
www.regouniversity.com

Continue to Get Resources and Stay Connected

1. Use RegoXchange.com for instructions and how-tos.
2. Talk with your account managers and your Rego consultants.
3. Connect with each other and Clarity experts at RegoGroups.com.
4. Sign up for webinars and join in-person Rego groups near you through at RegoConsulting.com
5. Join us for the next Rego University!