POSETTE 2025

Migration Consideration: Oracle to PostgreSQL

Neeta Goel
Founder, CEO & President

Sandeep Rajeev
Sales Director

newt|global
Database Modernization Acceleration Platform

Migration Considerations

newt|global
Database Modernization Acceleration Platform

Application & Interfaces

Schema Complexity

Infrastructure & Cloud

Database Performance

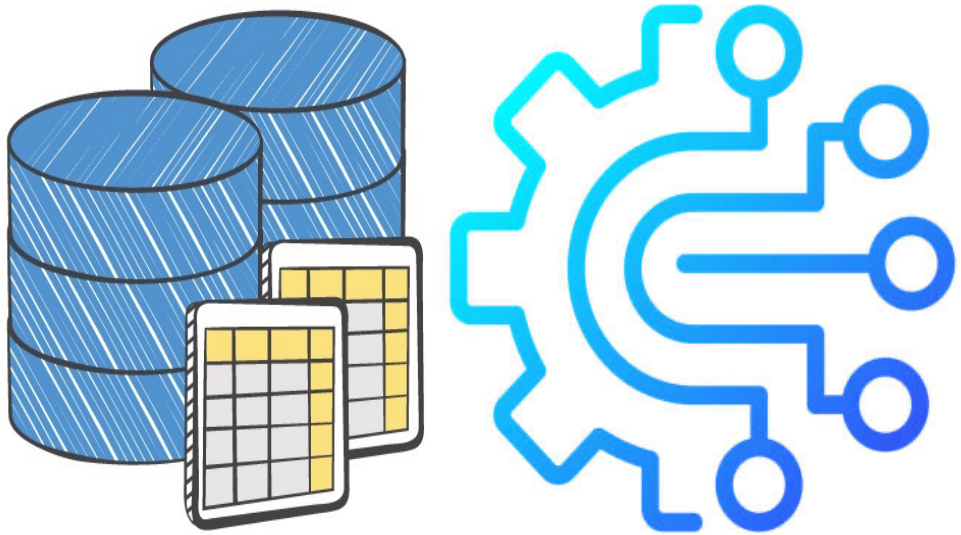
Batch Jobs & ETLs

Testing, Validation & Cloud Certification

Schema Conversion

newt|global
Database Modernization Acceleration Platform

- How do I convert PL*SQL? Stored Procedures, Functions, Triggers having business Logic.
- Special Features: Global Package Variables, Goto, Forall, Dynamic SQL, Pragma Autonomous Etc..



PLSQL Syntax: Handling Nulls Using Coalesce

newt|global
Database Modernization Acceleration Platform

Oracle

```
aa_ri_comm_regular = aa_ri_prem_regular + 1;
```

PostgreSQL

```
/* dmap converted statement start */
aa_ri_comm_regular := coalesce(aa_ri_prem_regular, 0) + 1;
```

Oracle implicitly handles NULL value with a math operator and variable will result into NULL

PostgreSQL: Assigning NULL + 1 to a numeric variable will result into a run time error and requires treatment for possible NULL value to ensure consistent behavior.

PLSQL Syntax: NULL vs Empty String

newt|global
Database Modernization Acceleration Platform

Oracle

```
IF pv_doccode IS NOT NULL THEN
```

PostgreSQL

```
/* dmap converted statement start */
if nullif(pv_doccode::text, '') is not null then
```

Oracle treats empty string in same way as NULL.

In PostgreSQL, explicit treatment is needed to handle empty string same as NULL.

Date Operations: trunc operator, Type Casting

newt|global
Database Modernization Acceleration Platform

Oracle

```
valuedt = trunc(to_date(aa_mast_row.incident_dt), 'dd/mm/yyyy'), 'mm')
```

PostgreSQL

```
/*  dmap  converted  statement  start  */
valuedt = date_trunc('month', to_date(aa_mast_row.incident_dt), 'dd/mm/yyyy'))
::timestamp without time zone
```

Oracle: trunc works for both date & numeric values.

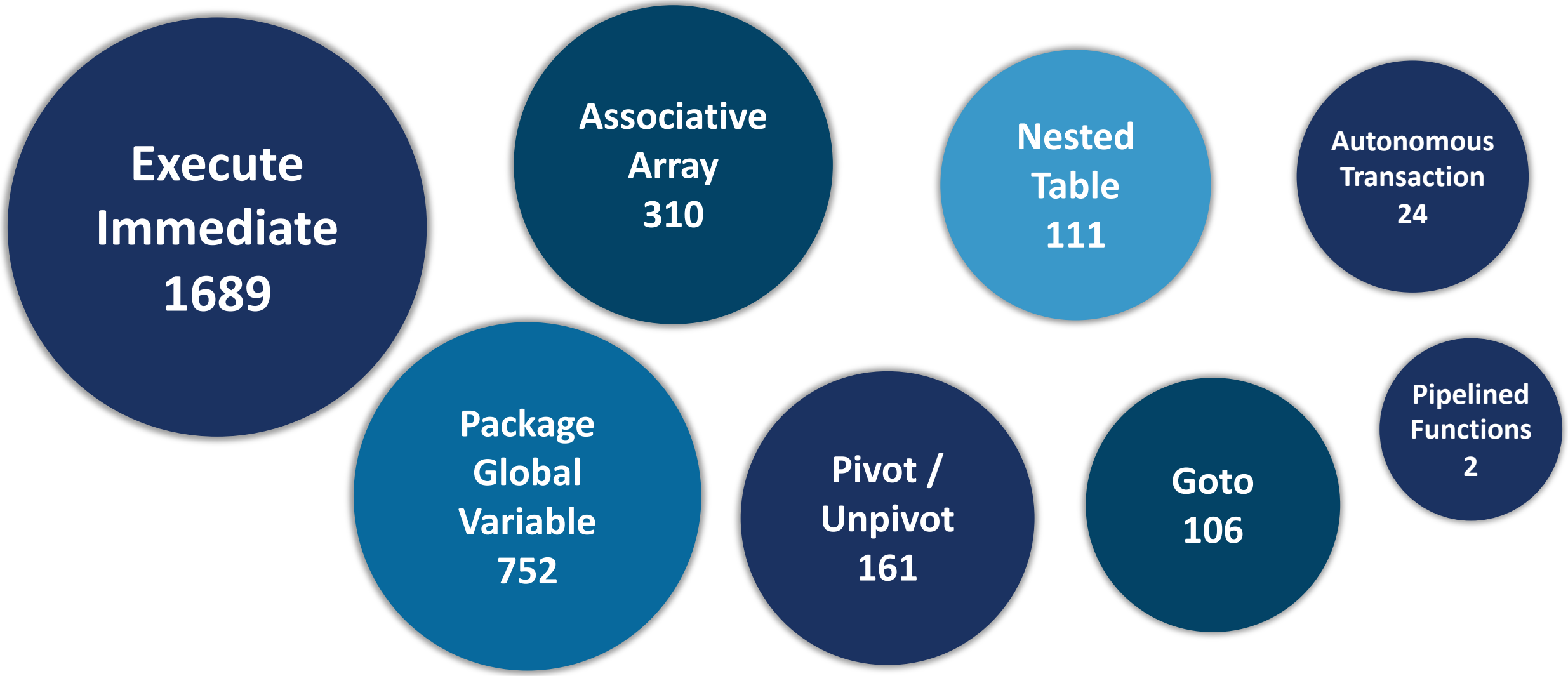
PostgreSQL: date_trunc and trunc are used for date and numeric resp. Additionally, if variable valuedt is a date without time component, type casting is required.

Schema Conversion: Special Objects

newt|global
Database Modernization Acceleration Platform

- Package Global Variable
- Associative Array
- Dynamic SQL (Execute Immediate)
- Forall
- Goto
- Save Exceptions
- Varray
- Autonomous Transaction
- Associative Array
- Nested Table
- Bulk Collect
- Synonyms (All Object Types)
- Dbms_Lob
- Pipelined Functions
- Pivot Or Unpivot
- Reverse Key Index
- EXTRACT XML
- Db Jobs
- Db Scheduler Jobs
- Compound triggers (with global variables)
- Oracle dictionary views (v$session, v$sql, dba_tables, dba_objects, dba_views etc.)
- Gather Stats
- Interval Partition
- Mview Refresh Methods
- Oracle Dict Views Ref
- Oracle Sys Dependency Obj
- SQL Loader Handling
- Utl_Smtp
- Utl_file

Illustration of Unsupported Oracle Features: One Schema



Dynamic SQL (Execute Immediate)

newt|global
Database Modernization Acceleration Platform

Oracle

```
EXECUTE IMMEDIATE
    'SELECT MCODE FROM ADMMONTH WHERE YCODE = :1'
INTO lv_mcode
USING PV_YCODE;
```

PostgreSQL

```
/*  dmap  converted  statement  start  */
EXECUTE 'SELECT mcode FROM admmonth WHERE ycode = ' || pv_ycode
INTO lv_mcode;
```

PostgreSQL handles dynamic SQL string directly using concatenation.

Handling Data Type Changes

newt|global
Database Modernization Acceleration Platform

Data Type (Oracle)	Data Type (PostgreSQL)	Occurrences (Table Columns)
NUMBER	Numeric	85152
FLOAT	Numeric	390
Int	Integer	208
DATE	Timestamp	20028
DATE	Timestamp without time zone	1500
TIMESTAMP(6)	Timestamp	17
VARCHAR2	Character Varying	73540
NVARCHAR2	Character Varying	12
BLOB	Bytea	60
CLOB	Text	40

Type Casting: Numeric

newt|global
Database Modernization Acceleration Platform

Oracle

```
aa_ri_prem_regular = ldc_rate * aa_epbfl_row.epb_qs/ 100 / 12 * ldc_exrate * ldc_loading;
```

PostgreSQL

```
/* dmap converted statement start */
aa_ri_prem_regular := ldc_rate * aa_epbfl_row.epb_qs::numeric / 100 / 12 *
ldc_exrate * ldc_loading;
```

Oracle has implicit data type conversions. aa_epbfl_row.epb_qs can be a text and contain a numeric value.

PostgreSQL is explicit in data type handling. A text or varchar column, must be type casted explicitly with ::numeric before using it in math operations.

Date Operations

newt|global
Database Modernization Acceleration Platform

Oracle

```
term_begin = add_months( last_day( lr_benf_mast_last.term_dt ) +1, -1 );
```

PostgreSQL

```
/* dmap converted statement start */
term_begin := add_months( last_day( lr_benf_mast_last.term_dt ) + INTERVAL '1 day', -1 );
```

Oracle handles +1 irrespective of Date type including date is timestamp with time zone.

PostgreSQL: +1 will work for date or timestamp without time zone. Requires explicit typecasting if date is timestamp with time zone.

Type Casting: Dates

newt|global
Database Modernization Acceleration Platform

Oracle

```
A_date => to_date('18/05/2016 16:15:45', 'dd/mm/yyyy hh24:mi:ss')
```

PostgreSQL

```
/*  dmap  converted  statement  start  */
A_Date => to_timestamp('18/05/2016 16:15:45', 'dd/mm/yyyy hh24:mi:ss')::timestamp
without time zone
```

Oracle: to_date supports both date and time implicitly. The date variable here is having timestamp without time zone.

PostgreSQL: to_date only handles dates. For Date with time, auto conversion to to_timestamp is done. Default return value from to_timestamp is with time zone, explicit typecasting is done to match date variable.

DB Server & Performance Consideration

newt|global
Database Modernization Acceleration Platform

Database Details

- Storage provisioned vs used
- Schema list by size
- Encrypted objects in database (Table columns, table spaces etc.)
- External tables, BLOB/CLOB & LONGRAW data columns
- Database parameters with non-default values

Server Parameters & Details

- CPU Cores, RAM, Oracle & OS Version
- Oracle licensing features provisioned & used
- Total instances in RAC & Data Guard setup
- DB properties such as Language, Time Zone, Character set etc.
- SGA & PGA Memory (Allocated, free and used)
- Active sessions and logons

AWR Trend & Performance Metrics

- AWR trend frequency & retention period
- CPU utilization (Min, Max & Avg.)
- Read & Write IOPS (Min, Max & Avg.)
- Read & Write Throughput (Min, Max & Avg.)
- Top CPU intensive SQL queries

Data Distribution & Characteristics

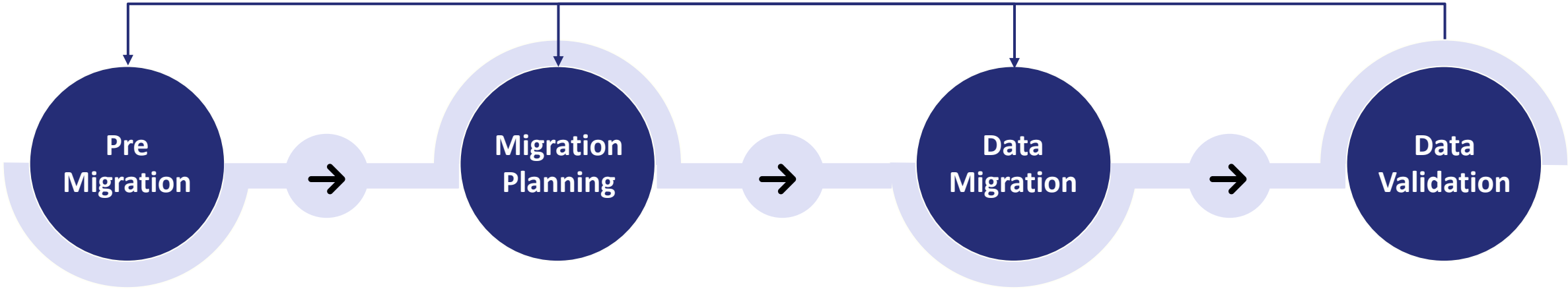
- Partition & Sub partition size & count
- Type of partitions (Range, Hash & Interval)
- Large tables greater than 100 GB
- LOB Size greater than 1 GB
- Logging & supplemental log data settings
- Redo log size

Data Migration

newt|global
Database Modernization Acceleration Platform

Iterative E2E Data Migration With DMAP

newt|global
Database Modernization Acceleration Platform



- Data Migration Planning
- Establish Validation Criteria
- Review & Update Server Parameters

- Source Data Validation
- Table Analysis & Defining Migration Treatment
- Define Grouping & Sequence for Data Migration

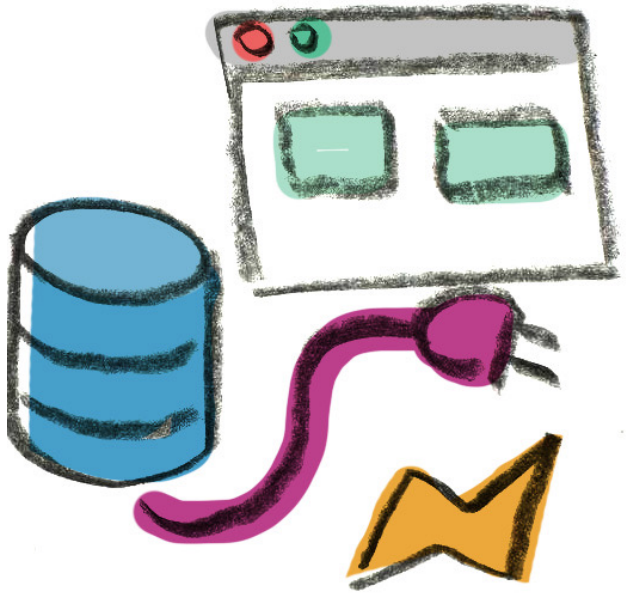
- Auto VM Creation, Config & Deployment
- Execute Data Migration & Monitor
- Auto Collection of Migration Stats
- Rerun of Migration
- Data Sync

- Physical Validation (Row Counts, Row Comparison)
- Logical Validation (Numeric, Character, Date Fields)

Who Migrates without apps?

newt|global
Database Modernization Acceleration Platform

- Embedded SQL Remediation (Static + dynamic Queries)
- Data Type Changes
- Procedure Changes
- Input/Output changes



Embedded Static SQL

newt|global
Database Modernization Acceleration Platform

Oracle

```
    String  sysDate = (String) session. createSQLQuery("SELECT
                        to_char(sysdate,'dd/mm/yyyy hh24:mi:ss' as datenow FROM dual")
          .addScalar("datenow", Hibernate. STRING) . uniqueResult();
    return sysDate;
}
```

PostgreSQL

```
    // DMAP Comment : Query remediated
    String sysDate = (String) session. createSQLQuery("SELECT
                        to_char(now(),'dd/mm/yyyy hh24:mi:ss' as datenow")
       .addScalar("datenow", Hibernate. STRING) .uniqueResult();
    return sysDate;
}
```

Embedded Dynamic SQL

newt|global
Database Modernization Acceleration Platform

Oracle

```
    sql.append(" SELECT TO_CHAR(TRUNC(SYSDATE) - ORDER_DATA, 'YYYYMMDD' )
AS CONDITION DATE ");
    sql.append(" FROM IP_MASTER_TABLE ");
    sql.append(" WHERE REF_TABLE = 'EMAIL_EXP' AND REF_KEY = 'L3' ");
    String smsNoList = (String) session. createSQLQuery(sql.toString())
        .addScalar("CONDITION_DATE", Hibernate. STRING) .uniqueResult();
    return smsNoList;
}
```

PostgreSQL

```
    // DMAP Comment : Query remediated
    sql.append(" SELECT TO_CHAR(date_trunc('day', now()) - ORDER_DATA, 'YYYYMMDD') AS
CONDITION_DATE ");
    sql.append(" FROM IP_MASTER_TABLE ");
    sql.append(" WHERE REF_TABLE = 'EMAIL_EXP' AND REF_KEY = 'L3' ");
    String smsNoList = (String)session.createSQLQuery(sql.toString())
        .addScalar("CONDITION_DATE", Hibernate.STRING).uniqueResult();
    return smsNoList;
}
```

Data Type Changes

newt|global
Database Modernization Acceleration Platform

Oracle

```
sql = "select * from tchfl " +
    " where tch_id = ? " ;
conn = getConnection();
stmt = conn. prepareStatement(sql);
stmt.setString(1, as_tch_id);
rs = stmt. executeQuery ( ) ;
while (rs.next()) {
    tch_id = rs.getString("tch_id");
    tch_dspval = rs.getString("tch_dspval");
    tch_dspval1 = rs.getString("tch_dspval1");
}
```

PostgreSQL

```
sql = "select * from tchfl " +
    " where tch_id = ? " ;
conn = getConnection();
stmt = conn. prepareStatement(sql);
stmt.setString(1, as_tch_id);
rs = stmt. executeQuery ( ) ;
while (rs.next()) {
    tch_id = rs.getString("tch_id");

// DMAP Comment: Data type change to int
    tch_dspval = rs.getInt("tch_dspval");
    tch_dspval1 = rs.getInt("tch_dspval1");
 }
```

Procedure / Function Calling Change

newt|global
Database Modernization Acceleration Platform

Oracle

```
connection = getSessionFactory() .getCurrentSession().connection();
plName = "{call GEN_REPORT_IPRUR001()}";

stmt = connection.prepareCall(plName);
stmt. execute();
```

PostgreSQL

```
Connection= getSessionFactory().getCurrentSession().connection();
// DMAP Comment : Query remediated
plName = “select GEN_REPORT_IPRUR001()“

stmt = connection.prepareCall(plName);
stmt.execute();
```

Input / Output Parameter

newt|global
Database Modernization Acceleration Platform

Oracle

```
cs.prepareCall("{ call wpnlfl_rtv(" +
                        "?,?," + "?" +
                        ")}");
cs.setString(1, getReSession().getSessionId());
cs.registerOutParameter(2, oracle.jdbc.OracleTypes.VARCHAR);
cs.registerOutParameter(3, oracle.jdbc.OracleTypes.CURSOR);
```

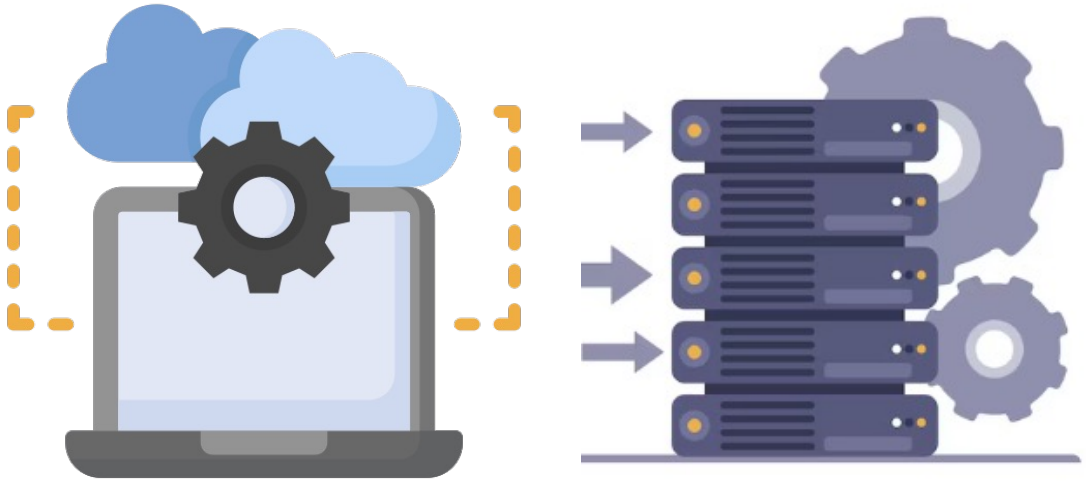
PostgreSQL

```
cs.prepareCall("{ call wpnlfl_rtv(" +
                        "?,?," + "?" +
                        ")}");
cs.setString(1, getReSession().getSessionId());

// DMAP Comment: Parameter type Changed
cs.registerOutParameter(2, java.sql.Types.VARCHAR);
cs.registerOutParameter(3, java.sql.Types.REF_CURSOR);
```

Let’s not forget Interfaces, Shell Scripts, Jobs…

newt|global
Database Modernization Acceleration Platform

Embedded SQL every where!!



PLSQL to PGSQL Auto Conversion

newt|global
Database Modernization Acceleration Platform

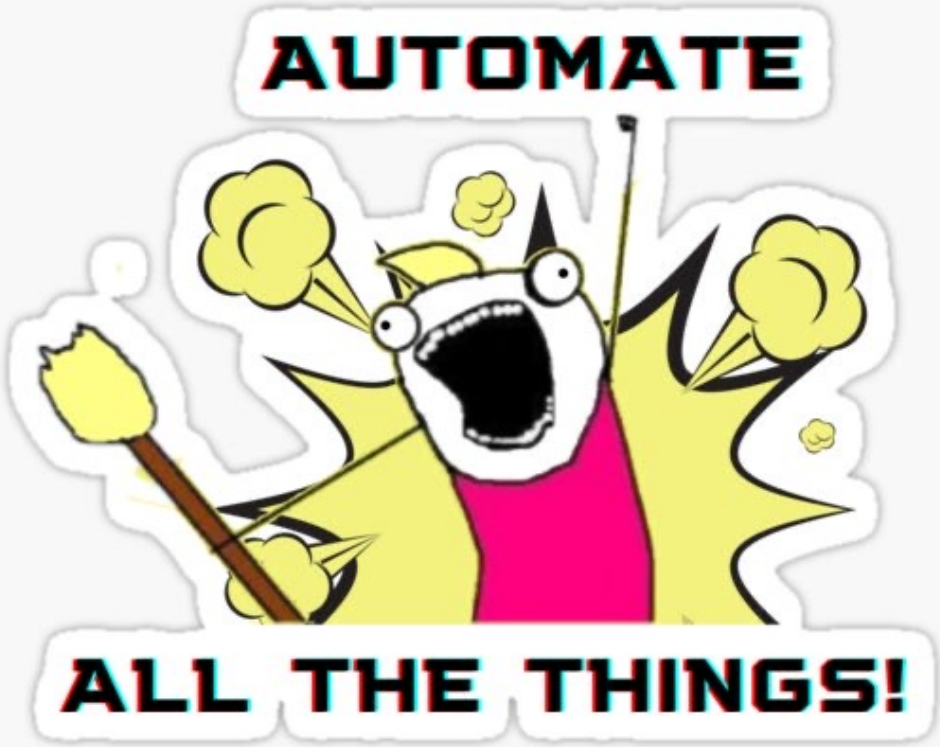
Oracle

```
f_db_status_inprogress_putMsg(){
    RESULT=`sqlplus -s /@$DBSERVER <<EOF
           set head off
           set feed off
           alter session set current_schema=pruris;
           UPDATE adj_clm_mast u
             SET u.up_st = '01'
                 ,u.adj_adate = sysdate
             WHERE u.up_st in ('00','01')
                   AND batch_no = '$batchNo';
           COMMIT;
           EXIT
           EOF`

    printlog "$RESULT"
   }
```

PostgreSQL

```
f_db_status_inprogress_putMsg(){
    RESULT=$(psql -t -q -A $DBSERVER <<EOF
           --set head off
           --set feed off
           SET search_path TO pruris, public, oracle,
   dmap_extension;
           UPDATE adj_clm_mast u
             SET up_st = '01'
                 ,adj_adate = now()::timestamp without
   time zone
             WHERE u.up_st in ('00','01')
               AND batch_no = '$batchNo';
           --COMMIT;
           --EXIT
           EOF)

    printlog "$RESULT"
}
```

Only Way



DMAP Led DB & Application Migration At Scale

newt|global
Database Modernization Acceleration Platform

APP DISCOVERY
- App (Java, .NET, C++) Source File Validity
- Source files having probable SQL queries

APP ASSESSMENT
- Embedded SQL Assessment for Java, .NET , C++ etc.
- Dead Code Analysis

APP CONVERSION
- ORM Neutral Embedded SQL Remediation (Static + dynamic Queries)
- Mapping Data Type changes

DEVELOPER EXPERIENCE
- Comments with TAGS for before and after code change blocks
- Audit trails for verified code changes

AT SCALE

Microsoft
OpenAI

AT SCALE

DISCOVERY
- DB Infra
- DB Size
- DB Objects
- DB LOC

ASSESSMENT
- Code/Data Objects
- Remediation Efforts
- TCO

SCHEMA CONVERSION
- Construct Conversion
- Report

SCHEMA VALIDATION
- Foreign & Primary Keys, Indexes
- Sequences
- Disabled Triggers
- Permissions

DATA MIGRATION
- Test Data
- Production Data

DATA VALIDATION
- Table Rows values & counts
- Arithmetic validation of numeric columns
- Validation of BLOB & CLOBs

Do I Look Like a Guy with a Plan?

Roadmap for DB & Application Modernization with Newt CMC™

newt|global
Database Modernization Acceleration Platform

App Team | DB Team | Infra Team | Security & Governance

One-Click Assessment

Schema Conversion & Validation

Data Migration & Validation

Application Remediation for Embedded SQL

Interface Remediation
Shell Scripts, Jobs, Reports, ETL Pipelines

Build & Deploy

Fix Errors

Testing: Dev . SIT . UAT
Performance . Security

Production
Cut-Over Planning

Production
GO/NO-GO

Production
Go-Live

Post-Production
Optimization

Infra Provisioning & Landing Zone Set up

Cloud Migration Cockpit (CMC™): Factory Model

newt|global
Database Modernization Acceleration Platform

Delivered 4x Faster
AI Ready
Savings from Oracle License

12 Months to 12 Weeks Guaranteed for a Mid Size DB & App to Production

PostgreSQL onboarding

Are you ready ?

Thank You