**Dual Schema TIN Validation System**

**Technical Reference Documentation**

**Table of Contents**

1. [Executive Summary](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#executive-summary)
2. [System Architecture](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#system-architecture)
3. [Installation Guide](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#installation-guide)
4. [Component Reference](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#component-reference)
5. [Usage Guide](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#usage-guide)
6. [Configuration](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#configuration)
7. [Troubleshooting](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#troubleshooting)
8. [Performance Guidelines](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#performance-guidelines)
9. [Security Considerations](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#security-considerations)
10. [Maintenance](https://claude.ai/chat/061b295a-eef6-4ed0-941e-3f69afeeb38e#maintenance)

**Executive Summary**

**Purpose**

The Dual Schema TIN Validation System is an Oracle PL/SQL solution designed to validate Taxpayer Identification Number (TIN) data between Legacy ALS systems and new Exadata platforms during ETL migration processes.

**Key Features**

* **Exact Data Replication**: Produces identical output to existing cK\_tin.sql script
* **Dual Schema Support**: Compares data between Legacy and Exadata schemas
* **Multiple Output Formats**: Summary, detailed comparison, single-schema validation
* **Self-Service Capability**: Business users can validate any TIN independently
* **Performance Optimized**: Hash-based comparison for accurate field-level matching

**Business Value**

* **Risk Mitigation**: Identifies data discrepancies before operational impact
* **Compliance Assurance**: Provides documented validation for audit requirements
* **Operational Efficiency**: Automated validation vs. manual processes
* **Migration Confidence**: Proves ETL process maintains data integrity

**System Architecture**

**Component Overview**

┌─────────────────────────────────────────────────────┐

│ Main Controller │

│ SP\_CK\_TIN\_DUAL\_SCHEMA │

└─────────────────┬───────────────────────────────────┘

│

┌─────────┴─────────┐

│ │

┌───────▼────────┐ ┌────────▼────────┐

│ Exadata │ │ Legacy │

│ Data Reader │ │ Data Reader │

│ (Ready) │ │ (Configurable) │

└───────┬────────┘ └────────┬────────┘

│ │

└─────────┬─────────┘

│

┌─────────▼─────────┐

│ Data Comparator │

│ & Hash Engine │

└─────────┬─────────┘

│

┌─────────▼─────────┐

│ Output Formatter │

│ (5 Format Types) │

└───────────────────┘

**Database Objects Structure**

**Object Types**

* t\_tin\_section\_data - Universal data structure for comparison
* t\_tin\_section\_data\_tab - Collection type for data records
* t\_comparison\_summary - Summary statistics structure
* t\_comparison\_summary\_tab - Collection type for summaries

**Functions**

* FN\_GENERATE\_SIMPLE\_HASH - Creates comparison hashes
* FN\_GET\_EXADATA\_TIN\_DATA - Extracts Exadata data
* FN\_GET\_LEGACY\_TIN\_DATA - Extracts Legacy data (customizable)
* FN\_COMPARE\_SCHEMAS - Performs data comparison

**Procedures**

* SP\_CK\_TIN\_DUAL\_SCHEMA - Main validation procedure
* SP\_TEST\_EXADATA\_EXTRACTION - Testing utility
* SP\_LEGACY\_SCHEMA\_HELP - Configuration guidance

**Installation Guide**

**Prerequisites**

* Oracle Database 11g or higher
* PL/SQL execution privileges
* Access to both Exadata and Legacy schemas
* DBMS\_OUTPUT package access

**Installation Steps**

**Step 1: Create Object Types**

-- Create universal data structure

CREATE OR REPLACE TYPE t\_tin\_section\_data AS OBJECT (

source\_schema VARCHAR2(20),

section\_name VARCHAR2(20),

tin\_number VARCHAR2(20),

tinsid NUMBER,

data\_fields VARCHAR2(4000),

record\_hash VARCHAR2(32),

raw\_data CLOB

);

/

CREATE OR REPLACE TYPE t\_tin\_section\_data\_tab AS TABLE OF t\_tin\_section\_data;

/

**Step 2: Create Utility Functions**

-- Hash function for comparison

CREATE OR REPLACE FUNCTION FN\_GENERATE\_SIMPLE\_HASH(p\_data VARCHAR2)

RETURN VARCHAR2 AS

v\_hash VARCHAR2(32);

BEGIN

SELECT SUBSTR(RAWTOHEX(UTL\_RAW.CAST\_TO\_RAW(p\_data || TO\_CHAR(SYSDATE, 'YYYYMMDDHH24MISS'))), 1, 32)

INTO v\_hash FROM DUAL;

RETURN v\_hash;

EXCEPTION

WHEN OTHERS THEN

RETURN TO\_CHAR(ABS(DBMS\_UTILITY.GET\_HASH\_VALUE(p\_data, 1, 999999999)));

END FN\_GENERATE\_SIMPLE\_HASH;

/

**Step 3: Deploy Main Components**

Execute the complete master script provided in the artifact.

**Step 4: Grant Permissions (if needed)**

GRANT EXECUTE ON SP\_CK\_TIN\_DUAL\_SCHEMA TO [ROLE\_NAME];

GRANT EXECUTE ON FN\_GET\_EXADATA\_TIN\_DATA TO [ROLE\_NAME];

**Step 5: Verify Installation**

-- Test installation

EXEC SP\_LEGACY\_SCHEMA\_HELP;

**Component Reference**

**SP\_CK\_TIN\_DUAL\_SCHEMA (Main Procedure)**

**Signature**

SP\_CK\_TIN\_DUAL\_SCHEMA(

p\_tin IN VARCHAR2,

p\_source\_schema IN VARCHAR2 DEFAULT 'BOTH',

p\_output\_format IN VARCHAR2 DEFAULT 'SIDE\_BY\_SIDE',

p\_legacy\_schema IN VARCHAR2 DEFAULT 'LEGACY\_SCHEMA'

)

**Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Type | Default | Description |
| p\_tin | VARCHAR2 | Required | TIN to validate (with or without dashes) |
| p\_source\_schema | VARCHAR2 | 'BOTH' | Data source: EXADATA, LEGACY, or BOTH |
| p\_output\_format | VARCHAR2 | 'SIDE\_BY\_SIDE' | Output format type |
| p\_legacy\_schema | VARCHAR2 | 'LEGACY\_SCHEMA' | Legacy schema name |

**Output Formats**

|  |  |  |
| --- | --- | --- |
| Format | Description | Use Case |
| EXADATA\_ONLY | Original cK\_tin.sql format from Exadata | Single schema validation |
| LEGACY\_ONLY | Original cK\_tin.sql format from Legacy | Single schema validation |
| SIDE\_BY\_SIDE | Parallel comparison view | Detailed analysis |
| SUMMARY | Statistical comparison report | Executive reporting |

**Data Sections Validated**

1. **ENT** - Entity/Taxpayer basic information
2. **TRANTRAIL** - Transaction trail records
3. **ENTMOD** - Entity modification history
4. **ENTACT** - Entity activity records
5. **TIMETIN** - Time and TIN relationship data

**FN\_GET\_EXADATA\_TIN\_DATA Function**

**Purpose**

Extracts TIN data from Exadata schema using exact logic from original cK\_tin.sql script.

**Signature**

FUNCTION FN\_GET\_EXADATA\_TIN\_DATA(p\_tin IN VARCHAR2)

RETURN t\_tin\_section\_data\_tab PIPELINED

**Query Logic**

* **ENT Query**: SELECT tin, tinsid, tpctrl, totassd, status, caseind, pyrind, casecode, subcode, assncff, assngrp, risk, arisk, extrdt FROM ent WHERE tin = v\_tptin
* **TRANTRAIL Query**: Joins with ENT, includes DECODE logic for dspcd field
* **ENTMOD Query**: Entity modification records with balance calculations
* **ENTACT Query**: Activity records with split rptcd fields
* **TIMETIN Query**: Time records with grade and risk comparisons

**Error Handling**

* Returns error record if TIN not found
* Handles data type conversion issues
* Provides meaningful error messages

**Hash-Based Comparison Engine**

**Algorithm**

1. **Data Normalization**: Converts all fields to comparable string format
2. **Hash Generation**: Creates MD5-equivalent hash for each record
3. **Field-Level Comparison**: Compares normalized field values
4. **Match Calculation**: Determines percentage match rates

**Hash Fields Format**

field1|field2|field3|...|fieldN

* Date fields: YYYYMMDD format
* Number fields: String representation
* Character fields: Trimmed values
* NULL values: Empty strings

**Usage Guide**

**Basic Usage Examples**

**Single Schema Validation**

-- Validate Exadata data only (replicates original cK\_tin.sql)

EXEC SP\_CK\_TIN\_DUAL\_SCHEMA('844607599', 'EXADATA', 'EXADATA\_ONLY');

-- Validate Legacy data only

EXEC SP\_CK\_TIN\_DUAL\_SCHEMA('844607599', 'LEGACY', 'LEGACY\_ONLY', 'YOUR\_LEGACY\_SCHEMA');

**Dual Schema Comparison**

-- Get summary comparison report

EXEC SP\_CK\_TIN\_DUAL\_SCHEMA('844607599', 'BOTH', 'SUMMARY', 'YOUR\_LEGACY\_SCHEMA');

-- Get detailed side-by-side comparison

EXEC SP\_CK\_TIN\_DUAL\_SCHEMA('844607599', 'BOTH', 'SIDE\_BY\_SIDE', 'YOUR\_LEGACY\_SCHEMA');

**SQL\*Plus Usage**

SET SERVEROUTPUT ON SIZE 1000000

SET LINESIZE 200

SET PAGESIZE 0

EXEC SP\_CK\_TIN\_DUAL\_SCHEMA('844607599', 'BOTH', 'SUMMARY');

**Testing and Validation**

**Installation Testing**

-- Test basic functionality

EXEC SP\_TEST\_EXADATA\_EXTRACTION('YOUR\_TIN');

-- Get configuration help

EXEC SP\_LEGACY\_SCHEMA\_HELP;

-- Verify TIN exists

SELECT COUNT(\*) FROM ent WHERE tin = YOUR\_TIN\_NUMBER;

**Data Validation Workflow**

1. **Identify Target TIN**: Select TIN for validation
2. **Test Exadata**: Run EXADATA\_ONLY format first
3. **Configure Legacy**: Set up legacy schema connection
4. **Run Comparison**: Execute SUMMARY format
5. **Analyze Results**: Review match percentages
6. **Investigate Differences**: Use SIDE\_BY\_SIDE for details

**Configuration**

**Legacy Schema Configuration**

**Required Customizations**

In FN\_GET\_LEGACY\_TIN\_DATA function, update these parameters:

-- Schema and Table Names

p\_legacy\_schema -- Your legacy schema name

'ent' -- Legacy ENT table name (if different)

'trantrail' -- Legacy TRANTRAIL table name (if different)

'entmod' -- Legacy ENTMOD table name (if different)

'entact' -- Legacy ENTACT table name (if different)

'timetin' -- Legacy TIMETIN table name (if different)

**Column Mapping**

If column names differ between schemas, update the field mappings:

-- Example: If legacy uses different column names

-- Replace 'tin' with 'taxpayer\_id' if that's your legacy column name

-- Replace 'tinsid' with 'entity\_id' if that's your legacy column name

**Database Links (if schemas are on different databases)**

-- Create database link if needed

CREATE DATABASE LINK legacy\_link

CONNECT TO legacy\_user IDENTIFIED BY legacy\_password

USING 'legacy\_database\_tns';

-- Update queries to use database link

FROM legacy\_schema.ent@legacy\_link

**Performance Tuning**

**Index Recommendations**

-- Ensure these indexes exist for optimal performance

CREATE INDEX idx\_ent\_tin ON ent(tin);

CREATE INDEX idx\_trantrail\_tinsid ON trantrail(tinsid);

CREATE INDEX idx\_entmod\_emodsid ON entmod(emodsid);

CREATE INDEX idx\_entact\_actsid ON entact(actsid);

CREATE INDEX idx\_timetin\_timesid ON timetin(timesid);

**Memory Configuration**

-- Set adequate DBMS\_OUTPUT buffer

DBMS\_OUTPUT.ENABLE(1000000);

-- For large datasets, consider increasing

DBMS\_OUTPUT.ENABLE(10000000);

**Troubleshooting**

**Common Errors and Solutions**

**ORA-00947: not enough values**

**Cause**: Object constructor parameter mismatch **Solution**: Verify all object type constructors have correct parameter count

-- Correct format

t\_tin\_section\_data(source\_schema, section\_name, tin\_number, tinsid, data\_fields, record\_hash, raw\_data)

**PLS-00382: expression is of wrong type**

**Cause**: Type conversion issue in assignments **Solution**: Explicitly construct object types

-- Correct assignment

v\_collection(v\_collection.COUNT) := t\_tin\_section\_data(param1, param2, ...);

**ORA-03113: end-of-file on communication channel**

**Cause**: Database connection lost or process crash **Solution**:

1. Restart database connection
2. Test with smaller datasets
3. Check for infinite loops
4. Verify TIN exists in database

**No Data Found**

**Cause**: TIN doesn't exist or wrong format **Solution**:

-- Verify TIN exists

SELECT COUNT(\*) FROM ent WHERE tin = YOUR\_TIN;

-- Check TIN format

SELECT DISTINCT LENGTH(TO\_CHAR(tin)), TO\_CHAR(tin) FROM ent WHERE ROWNUM <= 5;

**DBMS\_CRYPTO must be declared**

**Cause**: Missing database permissions **Solution**: Already resolved - script uses FN\_GENERATE\_SIMPLE\_HASH instead

**Debugging Steps**

**Step 1: Basic Connectivity**

SELECT SYSDATE FROM DUAL;

**Step 2: Table Access**

SELECT COUNT(\*) FROM ent;

SELECT COUNT(\*) FROM trantrail;

**Step 3: TIN Validation**

SELECT tin, tinsid FROM ent WHERE ROWNUM <= 5;

**Step 4: Function Testing**

SELECT COUNT(\*) FROM TABLE(FN\_GET\_EXADATA\_TIN\_DATA('VALID\_TIN'));

**Step 5: Incremental Testing**

-- Test each section individually

EXEC SP\_TEST\_EXADATA\_EXTRACTION('VALID\_TIN');

**Performance Guidelines**

**Expected Performance**

* **Single TIN Validation**: < 30 seconds
* **Small Dataset (< 100 records)**: < 60 seconds
* **Large Dataset (> 500 records)**: 2-5 minutes

**Performance Optimization**

**Database Level**

* Ensure proper indexing on TIN columns
* Gather table statistics regularly
* Monitor database resource usage

**Query Level**

* Use specific TINs rather than ranges
* Limit result sets when possible
* Consider parallel processing for batch operations

**Memory Management**

-- For large outputs, consider file-based output

-- Use chunked processing for very large datasets

**Monitoring Queries**

-- Check current session resource usage

SELECT \* FROM v$mystat WHERE statistic# IN (

SELECT statistic# FROM v$statname

WHERE name IN ('CPU used by this session', 'physical reads')

);

-- Monitor long-running operations

SELECT sql\_text, elapsed\_time FROM v$sql WHERE sql\_text LIKE '%SP\_CK\_TIN\_DUAL\_SCHEMA%';

**Security Considerations**

**Access Control**

* Grant execution privileges only to authorized users
* Use database roles for permission management
* Audit procedure execution if required

**Data Privacy**

* System accesses sensitive taxpayer data
* Ensure compliance with data protection regulations
* Log access for audit purposes

**Schema Security**

-- Example role-based security

CREATE ROLE tin\_validator\_role;

GRANT EXECUTE ON SP\_CK\_TIN\_DUAL\_SCHEMA TO tin\_validator\_role;

GRANT tin\_validator\_role TO authorized\_user;

**Maintenance**

**Regular Maintenance Tasks**

**Monthly**

* Review execution logs for errors
* Update table statistics
* Monitor performance metrics

**Quarterly**

* Validate against sample TINs
* Review and update documentation
* Performance tuning analysis

**Annual**

* Complete system review
* Update configuration as needed
* Archive old execution logs

**Version Control**

* Track all configuration changes
* Document customizations
* Maintain rollback procedures

**Backup Procedures**

-- Export procedure definitions

expdp system/password schemas=YOUR\_SCHEMA include=PROCEDURE,FUNCTION,TYPE

**Appendix**

**Complete Parameter Reference**

**SP\_CK\_TIN\_DUAL\_SCHEMA Parameters**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter | Required | Type | Values | Description |
| p\_tin | Yes | VARCHAR2 | Any valid TIN | Target TIN for validation |
| p\_source\_schema | No | VARCHAR2 | EXADATA, LEGACY, BOTH | Data source selection |
| p\_output\_format | No | VARCHAR2 | EXADATA\_ONLY, LEGACY\_ONLY, SIDE\_BY\_SIDE, SUMMARY | Output format type |
| p\_legacy\_schema | No | VARCHAR2 | Schema name | Legacy database schema |

**Error Code Reference**

| **Error Code** | **Description** | **Resolution** |
| --- | --- | --- |
| ORA-00947 | Not enough values | Check object constructors |
| PLS-00382 | Wrong type | Explicit type conversion |
| ORA-03113 | Connection lost | Restart connection |
| ORA-01403 | No data found | Verify TIN exists |

**Sample Output Formats**

**SUMMARY Format Example**

=== VALIDATION SUMMARY REPORT ===

SECTION | EXADATA | LEGACY | MATCHES | DIFF | MATCH%

------------|---------|--------|---------|------|-------

ENT | 1 | 1 | 1 | 0 | 100%

TRANTRAIL | 15 | 15 | 15 | 0 | 100%

ENTMOD | 25 | 25 | 25 | 0 | 100%

ENTACT | 38 | 38 | 38 | 0 | 100%

\*\*\* OVERALL MATCH PERCENTAGE: 100% \*\*\*

✓ VALIDATION PASSED: All data matches perfectly between schemas!

**Support and Contacts**

**Technical Support**

* **Database Issues**: Contact your DBA team
* **Configuration**: Use SP\_LEGACY\_SCHEMA\_HELP procedure
* **Performance**: Monitor using provided queries

**Documentation Version**

* **Version**: 1.0
* **Last Updated**: Current Date
* **Next Review**: Quarterly

*This document serves as the complete technical reference for the Dual Schema TIN Validation System. Keep this document updated as the system evolves and configurations change.*