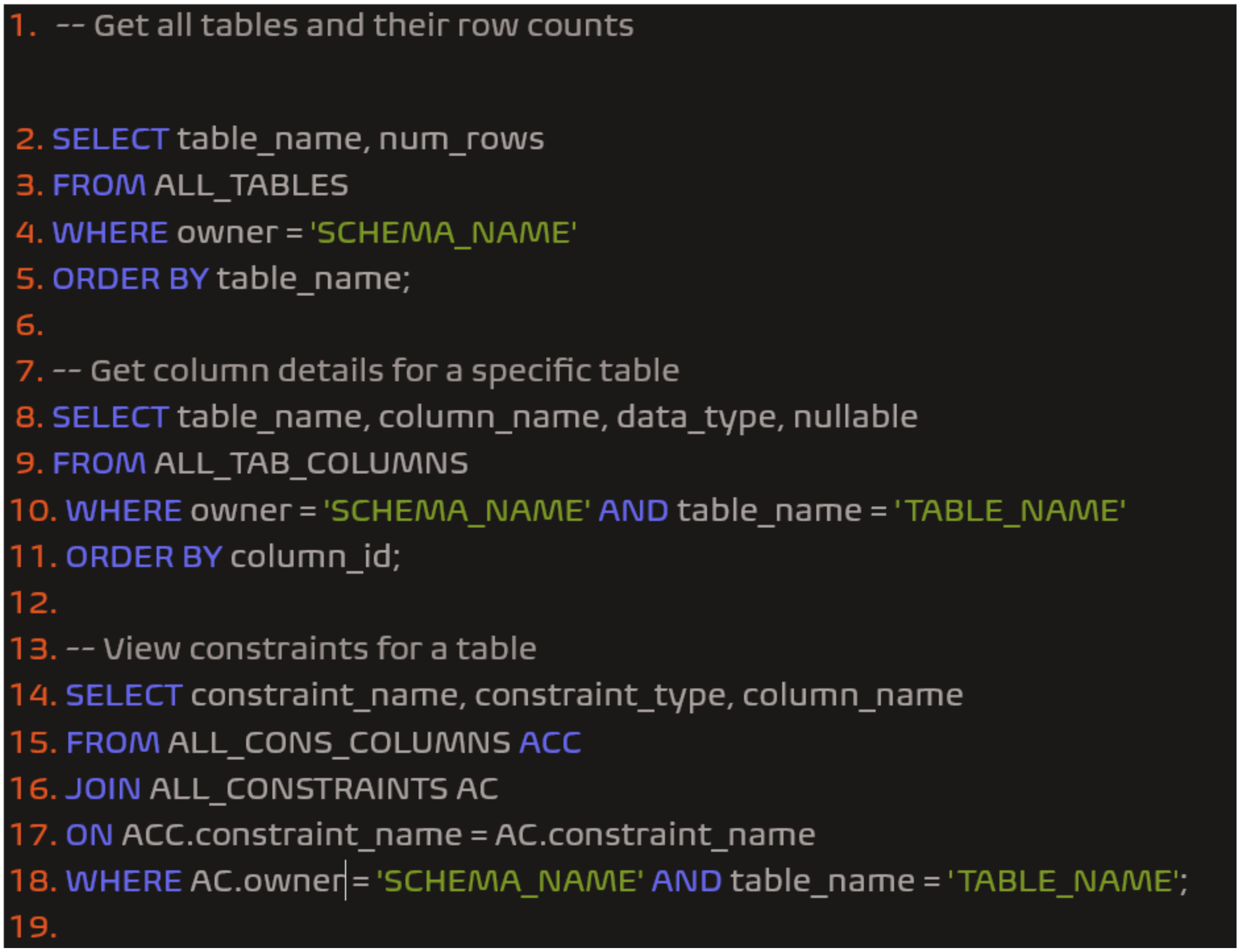
10 Critical Steps You Must Know as a New Oracle DBA

Asfaw Gedamu

Just Landed Your First Oracle DBA Role? Here’s How to Hit the Ground Running!



```
-- Get all tables and their row counts

SELECT table_name, num_rows
FROM ALL_TABLES
WHERE owner = 'SCHEMA_NAME'
ORDER BY table_name;

-- Get column details for a specific table
SELECT table_name, column_name, data_type, nullable
FROM ALL_TAB_COLUMNS
WHERE owner = 'SCHEMA_NAME' AND table_name = 'TABLE_NAME'
ORDER BY column_id;

-- View constraints for a table
SELECT constraint_name, constraint_type, column_name
FROM ALL_CONS_COLUMNS ACC
JOIN ALL_CONSTRAINTS AC
ON ACC.constraint_name = AC.constraint_name
WHERE AC.owner = 'SCHEMA_NAME' AND table_name = 'TABLE_NAME';
```

You’ve been handed bulky operation and maintenance manuals, and it all feels overwhelming. Where do you even start?

With over 10 years of experience managing Oracle databases that store the profile and transaction data of over 70 million customers, I know exactly how you feel. That’s why I’ve created this guide to help you get started.

By breaking it down into **10 critical steps**, you’ll be ready to tackle your new role with confidence and set yourself up for success.

1. Assess Instance Information

```sql
SELECT instance_name, version, status
FROM v$instance;
```

Why?

- This ensures you know the database version, instance name, and status. It helps verify compatibility and that the instance is running properly.
- Know your database version, instance name, and status.
- Why: This is your starting point to understand the environment you're managing.

2. Check Database Storage

```sql
SELECT tablespace_name,
       ROUND(SUM(bytes) / (1024 * 1024), 2) AS total_space_mb,
       ROUND(SUM(free_bytes) / (1024 * 1024), 2) AS free_space_mb
FROM DBA_TABLESPACES
JOIN DBA_FREE_SPACE USING (tablespace_name)
GROUP BY tablespace_name
ORDER BY tablespace_name;

```

Why?

- Ensure you have enough storage for current operations and future growth.
- Running out of space can lead to downtime and performance issues.

3. Validate User Accounts

```sql
SELECT username, account_status, default_tablespace, temporary_tablespace
```

```
FROM dba_users;

```

Why?

- Identify locked or unused accounts and assign appropriate roles and resources.
- Securing user accounts minimizes security risks and ensures proper access control.

4. Verify Backup Configurations

```
SHOW PARAMETER control_files;
SELECT * FROM v$backup;

```

Why?

- Confirm that backups are properly configured and tested for recovery.
- Why: Backups are your safety net in case of data loss or corruption.

5. Set Up Auditing and Logging

```
SELECT audit_trail
FROM dba_audit_trail;

```

Why?

- Enable auditing for compliance and monitoring of critical activities.
- Auditing helps track changes and detect unauthorized access.

6. Explore Schema Metadata

a. List All Tables:

```sql
SELECT table_name, num_rows, last_analyzed
FROM ALL_TABLES
WHERE owner = 'SCHEMA_NAME'
ORDER BY table_name;

```

b. View Column Details:

```sql

SELECT table_name, column_name, data_type, nullable
FROM ALL_TAB_COLUMNS
WHERE owner = 'SCHEMA_NAME'
ORDER BY table_name;

```

c. Review Constraints:

```sql
SELECT constraint_name, constraint_type, column_name
FROM ALL_CONS_COLUMNS ACC
JOIN ALL_CONSTRAINTS AC
ON ACC.constraint_name = AC.constraint_name
WHERE AC.owner = 'SCHEMA_NAME';

```

Why?

- Schema exploration provides an understanding of table structures, relationships, and constraints.
- Understand table structures, relationships, and constraints.
- Knowing your schema is essential for troubleshooting and optimization.

7. Inspect Performance Metrics

```
SELECT *
FROM v$sysmetric
WHERE metric_name IN ('Database Time Per Sec', 'CPU Usage Per Sec');

```

Why?

- Identify and address performance bottlenecks proactively.
- A well-tuned database ensures smooth operations and user satisfaction.

8. Check Tablespace Status

```
SELECT file_name, tablespace_name, status
FROM dba_data_files;

```

Why?

- Verify that tablespaces and data files are online and functioning correctly.

- Tablespaces are the backbone of your database storage—keep them healthy!

9. Verify High Availability (HA) Configurations

```
SELECT name, value
FROM v$parameter
WHERE name LIKE '%dg%';

```

Why?

- Ensure disaster recovery and failover mechanisms are in place.
- HA configurations minimize downtime and ensure business continuity.

10. Monitor Alerts and Logs

```
SELECT message_text, timestamp
FROM v$alert_log
WHERE timestamp > SYSDATE - 7;

```

Why?

- Stay ahead of potential issues by regularly reviewing alerts and logs.
- Proactive monitoring helps you catch and resolve issues before they escalate.
- Reviewing recent alerts helps identify errors or warnings before they escalate.

Summary

These steps ensure you assess the database's current state, address immediate risks, and prepare for long-term management. Each step prioritizes security, performance, and reliability, which are essential for any DBA role. By exploring schema metadata as a core component, you'll have a thorough understanding of the database structure to support day-to-day operations effectively.

💡 Pro Tip: Document everything! Keeping detailed records of configurations, changes, and issues will save you time and effort in the long run.

🔗 Are you using these best practices? Share your tips, experiences, or questions in the comments below! Let's learn and grow together as a community of DBAs.