Creating Master Encryption Keys for PDBs

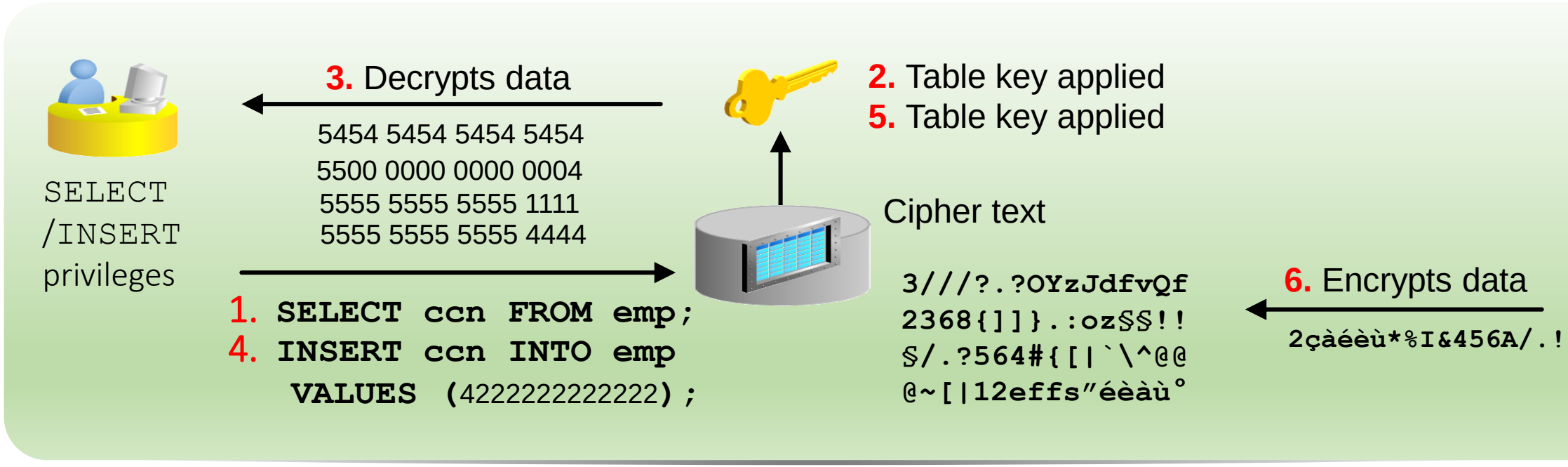
Objectives

- After completing this lesson, you should be able to:
  - Describe the implementation of master encryption keys for PDBs
  - Create and activate a master encryption key for a new PDB

Encryption in Database Cloud Service

- Database Cloud Service databases include a key management framework that stores and manages keys and credentials used to encrypt data in the database data files and in backups.
- The key management framework includes:
  - The keystore (referred to as a wallet in Oracle Database 11*g* and previous releases) to securely store Transparent Data Encryption (TDE) master encryption keys
  - The management framework to securely and efficiently manage the keystore and key operations for various database components

Transparent Data Encryption (TDE): Overview

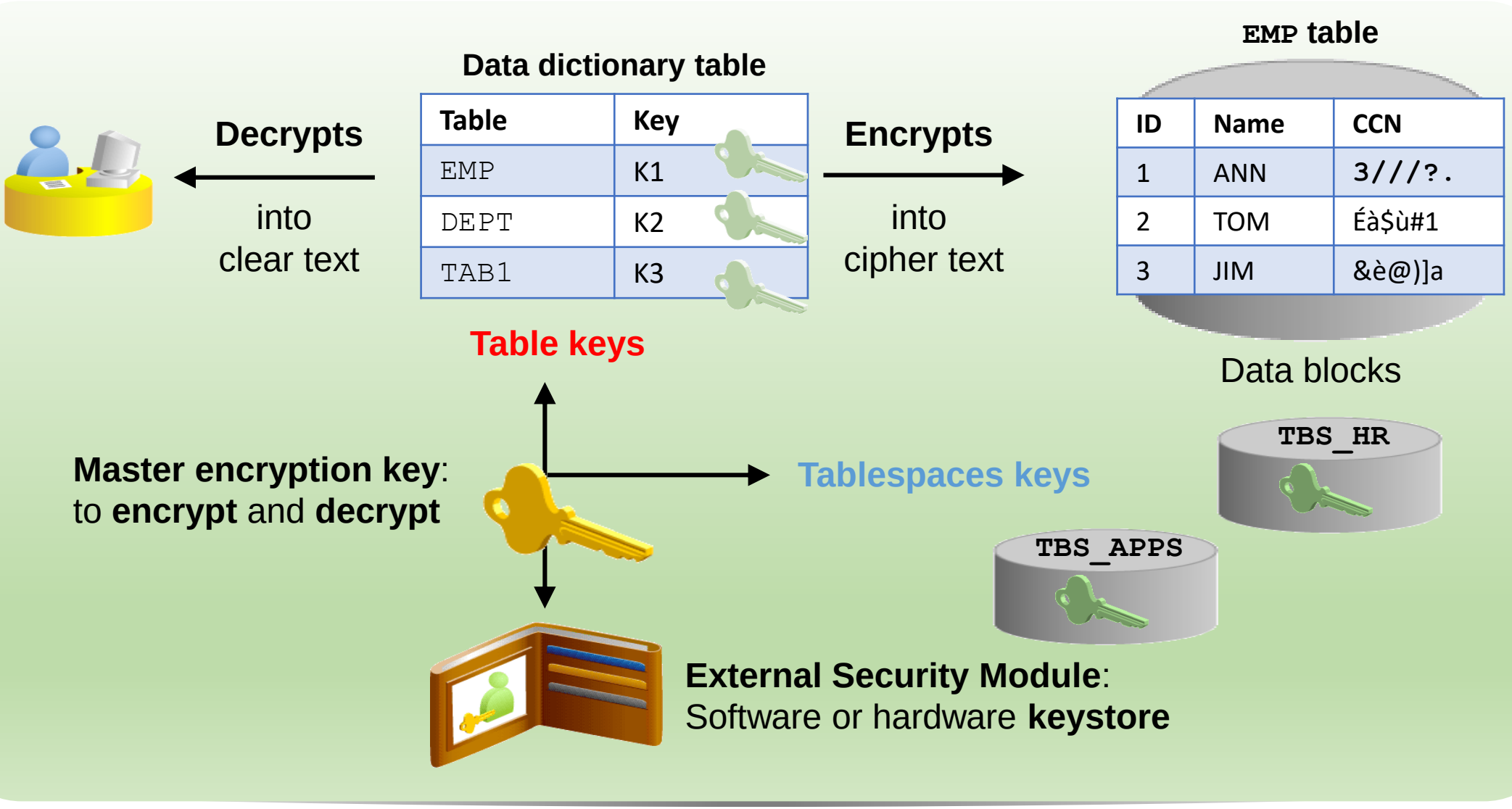


- Encrypts data in data files, redo log files, archived redo log files, and backup files
- Encrypts data in memory (only for column encryption)
- Manages keys automatically
- Does not require application changes

Components of TDE

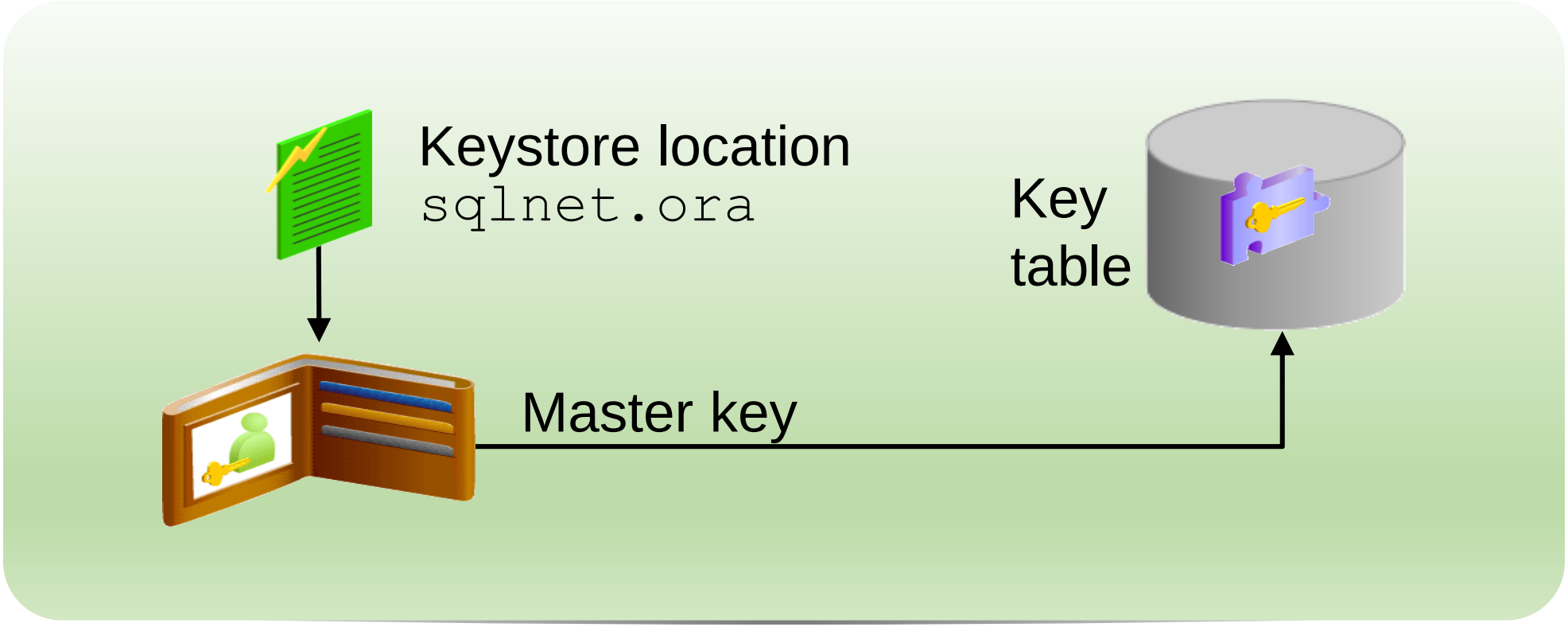
- Key architecture
  - Two-tier architecture: A unified master encryption key stored in an external security module is used to encrypt the table key or tablespace key.
  - Low overhead re-key operation: Some security regulations require periodical changes of encryption keys.
- External security module
  - Software keystore
  - Hardware keystore (HSM)
- Algorithm support

Using TDE


Decrypts
into
clear text
Data dictionary table
Table
Key
EMP
K1
DEPT
K2
TAB1
K3
Encrypts
into
cipher text
EMP table
ID
Name
CCN
1
ANN
3///?.
2
TOM
Éà$ù#1
3
JIM
&è@)]a
Data blocks
Table keys
Master encryption key:
to encrypt and decrypt
Tablespaces keys
TBS_HR
TBS_APPS
External Security Module:
Software or hardware keystore

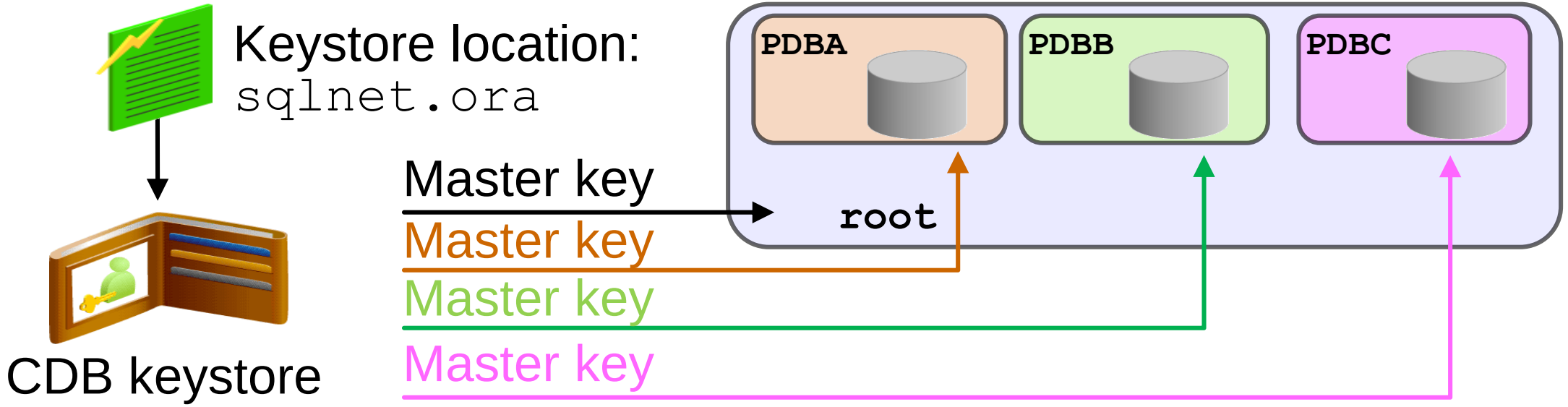
Defining the Keystore Location

- The location of the keystore file is specified in an entry in the `$ORACLE_HOME/network/admin/sqlnet.ora` file.



CDB and PDB Master Encryption Keys

- There is one master encryption key per PDB to encrypt PDB data.
- The master key must be transported from the source database keystore to the target database keystore when a PDB is moved from one host to another.
- After creating or plugging in a new PDB, you must create and activate a master encryption key for the PDB.



Do You Need to Create and Activate a Master Encryption Key?

1. Set the container to the PDB.

2. Query the `STATUS` column in `V$ENCRYPTION_WALLET`.

```
SQL> ALTER SESSION SET CONTAINER = pdb;
```

3. If `STATUS` contains a value of `OPEN_NO_MASTER_KEY`, create and activate the master encryption key (shown in the next slide).

```
SQL> SELECT wrl_parameter, status, wallet_type
  2  FROM v$encryption_wallet;
```

Creating and Activating a Master Encryption Key

1. Close the auto-login keystore in the root container and then reopen it as a password keystore.

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE close;
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE open
  2  IDENTIFIED BY keystore-password CONTAINER = all;
```

2. Set the container to the PDB. Create and activate a master encryption key in the PDB.

```
SQL> ADMINISTER KEY MANAGEMENT SET KEY USING TAG 'tag'
  2  IDENTIFIED BY keystore-password
  3  WITH BACKUP USING 'backup_identifier';
```

3. Query `V$ENCRYPTION_WALLET` again, verifying that `STATUS` is `OPEN`.

```
SQL> SELECT wrl_parameter, status, wallet_type
  2  FROM v$encryption_wallet;
```

Summary

- In this lesson, you should have learned how to:
  - Describe the implementation of master encryption keys for PDBs
  - Create and activate a master encryption key for a new PDB

Practice 11: Overview

- 11-1: Creating and Activating an Encryption Key
- 11-2: Creating and Activating the Encryption Key for PDB2