Week2_BackupRestore_SecurityUserManagement

Managing Databases

Backup and Restore Databases

Scenarios:

- Saving a copy of data for protection
- Recovering from data loss
    - After unplanned shutdown
    - Accidental deletion
    - Data corruption
- Move to a different database system
- Share data with business partners
- Use a copy of the data, e.g dev or test
-

Physical vs. logical backups

1. Logical backup
    - Contains DDL and DML commands to recreate the database
    - Can reclaim wasted space
    - Slow and may impact performance
    - Granular
    - `Backup/restore`, `import/export`, `dump/load` utilities
2. Physical backup:
    - Copy of physical files, including logs and configuration
    - Smaller and quicker
    - Less granular
    - Can only restore to similar RDBMS
    - Common for specialized storage and Cloud

You can backup a whole database, schema, tables, subset of data or other objects

Key considerations

- Check taht your backup is valid
- Check that your restore plan works
- Ensure that your back up files is secure

Backup options

1. Compression:
    - Reduces size for storage and trasmission
    - Increases time for backup and restore process
2. Encryption:
    - Reduces the risk of data being compromised
    - Increases time for backup and restore process

Types of Backup

- **Full backup**:
    - Backs up all the specified data
    - Multiple copies of the backup means storing many instances of a large file
    - Only storing one copy risks data loss if file is corrupt
    - Could be needlessly backing up unchanging data
    - Must secure backup files

Full backups are simple to create and restore, but can be slow to run and result in large files

- **Point-in-time recovery**:
    - Uses logged transatctions to restore to an earlier point in time
    - Enables you to recover the data to the state it was in a particular time
      Provides a more granular recovery model than just using full database backups
- **Differential backups**
    - A copy of any data that has changed since the last full backup was taken
    - You can perform a full backup on Sunday, and run differential backup each weekday
      Are quicker to run than full backups, but the restore process can take longer
- **Incremental backups**
    - A copy of any data that has changed since the last backup of any type as taken
      Are even quicker to run, but the restore process can take even longer

Back up policies

- **Hot backup** - taken while data is in use (also called online backups)
- User can continue with the activities while backing up

- Can impact in performance degradation for users when backup is running
- Can present data integrity issues if data is changed while in backup
- **Cold backup** - data is offline
- Impact data availability
- Eliminates data integrity risks
- Can not be used in 24/7 environments

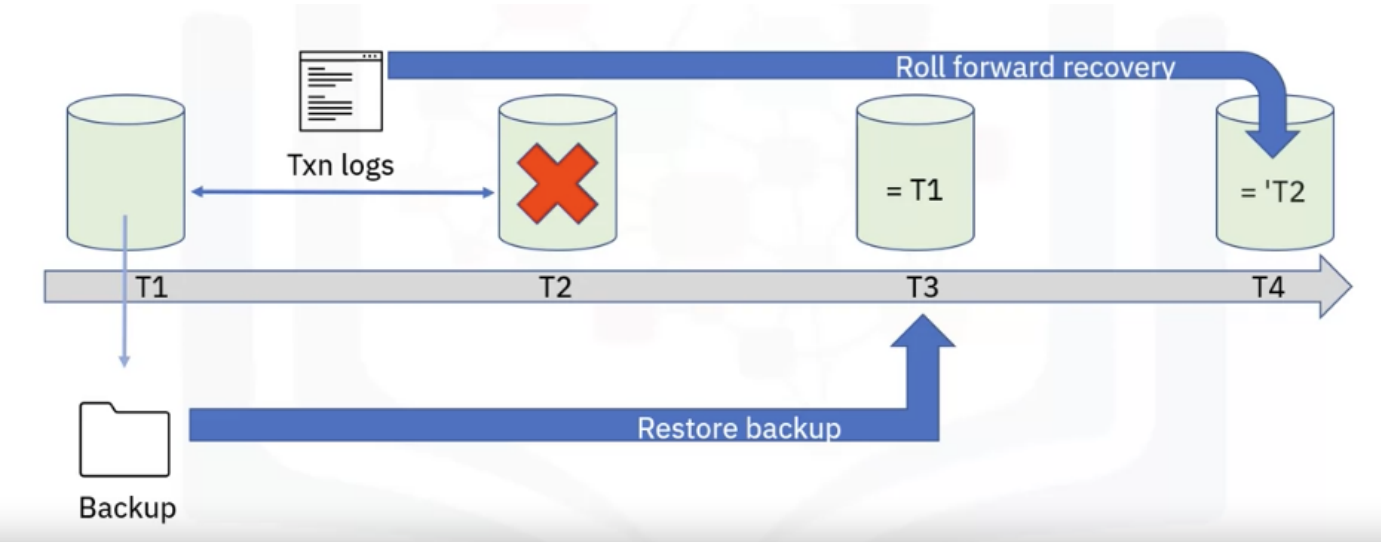
Considerations

Your backup policy should be determined from your recovery needs and your data usage. Most managed cloud databases provide automated backup functionality.

- Frequency:
  - Is data regularly changing or being added?
  - Is the existing table large?
- Schedule:
  - Is the data accessed equally across the 24-hour day
  - Is it accessed at weekends?
- Automated

Using Database Transaction Logs for Recovery

A database management system (DBMS) uses transaction logs to keep track of all transactions that change or modify the database. The information stored in these transaction logs can be used for recovery purposes when data is accidentally deleted, or a system or hardware failure such as a disk crash occurs. In case of such events, the transaction log, in conjunction with a database backup, can help restore your database back to a consistent state at a specific point in time.



A recommended best practice is to Isolate logs on different volumns from data
- Increases performance

- Recoverability

Log mirroring - store second copy of log files in an alternative location
Log shipping - copy and send logs to replica or standby servers

Log files are stored separately from database files by changing configuration settings and can be accessed using command-line or graphical user interfaces

Log files contain transaction ID number, databse record type, log sequence number, and other information

Security and User Management

When securing a db, you need to consider the security of the server and operating system, as well as the db and data. Also, users need to be authenticated on the server or the database to access it. Adter authenticated, users need to be authorized to access the objects and data in the database.
Consider principles like principle of least privilage, and monitoring and auditing database acitivity. Encryption can strenghten your data security, and dont forget to secure your applications.

Server security

- On premise servers:
  - who has access?
  - how are they physically secured?
- Manage cloud:
  - check provider documentation

Operating system configuration

- Regular patching
- system hardening
- access monitoring

RDMS configuration

- Regular patching
- Review and use system-specific security features
- Reduce the number of admins

Accessing db and objects

Authentication

- Similar to:
  - PIN for cell phone
  - Password for computer

- Verifies that the user is who they claim to be
    - validates username and password
- External authentication methods:
    - PAM
    - Windows login IDs
    - LDAP
    - Kerberos

Authorization

- Authorized to access:
    - objects
    - data
- Grant privileges to:
    - Users
    - Groups
    - Roles

Privileges

- Allow users to select, update, insert, delete, alter table structure, etc
- Apply the principle of the least privilege
- Monitor:
    - who access what objects
    - what actions they perform
- Audit
    - Actual access against security plan
- Encryption
    - Adds another layer of security
    - Intruders need to decrypt
    - Industry & regional regulations:
        - Algorithms
        - Key management
    - Performance impact

Application security

- SQL injection strings
- Insecure code

Users, groups & roles

Groups

Groups and roles represent collections of users that perform similar functions, or have a similar status in an organization. Examples of groups are Employees, Developers, or Sales Personnel. Members of groups can be users and other groups. When users log on, they cannot select a group they want to use for a session. They always log on with all the permissions associated with the groups to which they belong.

A group is a collection of users with a given set of permissions assigned to the group (and transitively, to the users). A role is a collection of permissions, and a user effectively inherits those permissions when he acts under that role.

Users can become members of groups and roles and groups and roles defined in authentication providers. A user can belong to one or more groups or roles. If users are members of more than one group, their access permissions are merged.

You can create users directly in a database system or map them to users in the operating system

In some systems, you define groups in the database to manage users
And in others, you can map database groups to operating system groups
You can use predefined database roles to assign privileges to common sets of database users
You can define custom roles for your own requirements
Gorups and roles simplify user management

Authorization

You grant permissions to users, groups or roles
Permissions control access to databases and the objects in them
The range of permissions allow for fine tuning of database access
You can revoke a previously granted permission and deny a permission to override an existing granted permission

Database Access

```
GRANT CONNECT TO 'salesteam'
```

```
GRANT CONNECT TO 'user'
```

```
REVOKE CONNECTION TO `public`
```

```
GRANT SELECT ON mydb.mytable TO 'salesteam'

GRANT INSERT ON mydb.mytable to 'salesteam'

GRANT UPDATE ON mydb.mytable TO 'salesteam'

GRANT DELETE ON mydb.mytable TO 'salesteam'
```

```
GRANT CREATE TABLE TO 'salesteam'

GRANT CREATE PROCEDURE TO 'user'
```

```
GRANT VIEW ON mydb.myproc TO 'salesteam'

GRANT EXECUTE ON mydb.myproc TO 'salesteam'

GRANT ALTER ON mydb.myproc TO 'salesteam'
```

Revoke and deny access

```
REVOKE SELECT ON mydb.mytable TO 'salesteam'
```

overwrites all permission

```
DENY SELECT ON mydb.mytable to 'salesteam'
```

Auditing database activity

- Auditing does not directly protect your database, but does identify gaps in your security
- Some industries, customers, or regions may require audit logs are cerated an retained
- You should audit both successful and failed attempts to access your database
- You should audit and review all activity in your database

Encrypting Data

Why?

- Provide another layer in security system
- Often last line of defense
- Can protect data:
  - at rest
  - during transmission

- May be required by:
    - Industry
    - Region
    - Customer

Protecting data at rest

- Transparent data encryption (TDE) - entire database
- There's also encryption at table and column level - but it can show performance issue and complex set up

Algorithms and keys

- **Symmetric encryption:**
    - Same key is used to encrypt and decrypt. Examples: aes, des. Key is shared with all users
    - Increases the likelihood of compromise
- **Asymmetric encryption**
    - Uses two keys: one public, one private
    - Public key encrypts, private key decrypts
    - Examples: RSA, ECC
- **Transparent data encryption**
    - Encryption and key management
    - Not visible to users
    - Database engine encrypts and decrypts data
    - Also encrypts backups
- **Customer managed keys**
    - Provide the data owner with more control over their data stored in the cloud
    - Bring your own Key (BYOK)
    - Cloud provider responsible for encryption process
    - You remain responsible for key management
    - Cloud provider cannot access your confidential data
    - Security admins manage keys; database admins manage data
    - Complete control over keys and their lifecycle
-

Encryption vs performance

- Choice of symmetric or asymmetric encryption may be configurable

- All encryption takes time and effort
- Asymmetric algorithms generally use longer keys, therefore have greater overheads
- Symmetric algorithms are often sufficient

Protecting data in transit

- Often provided by RDBMS
- Protocols:
  - TLS
  - SSL
- May encrypt by default, may be configurable
- Performance impact
