

Majority Database Migrations

Custom checksum journal leveraging dbup

[Link to Pull Request](#)

Table of Contents

1. How It Works Today
2. Custom Checksum Journaling
3. Demo
4. Discussion
5. Summary After Discussion

1. How It Works Today

Current Process

- Add a SQL script to a folder called `/Migrations` .
- The script is executed once and never runs again.

- Minority.Wallet.Db
 - Dependencies
 - Migrations
 - 001_AccountHolder_CreateTable.sql
 - 002_AccountHolderInsert_SP.sql
 - 003_AccountHolderGetByAccountHolderId_SP.sql
 - 004_Account_CreateTable.sql
 - 006_AccountInsert_SP.sql
 - 007_AccountGetByAccountId_SP.sql
 - 008_AccountGetAllByAccountHolderId.sql
 - 009_VirtualCardStatus_CreateTable.sql
 - 010_PhysicalCardStatus_CreateTable.sql
 - 011_VirtualCard_CreateTable.sql
 - 012_VirtualCardInsert_SP.sql
 - 013_VirtualCardUpdateUpdate_SP.sql
 - 014_VirtualCardGetByVirtualCardId_SP.sql
 - 015_VirtualCardGetAllByAccountId_SP.sql
 - 016_AccountHolderProfile_CreateTable.sql
 - 017_AccountHolderProfileInsert_SP.sql
 - 018_AccountHolderProfileUpdate_SP.sql
 - 019_AccountHolderProfileGetByAccountHolderId_SPs.sql
 - 020_Transaction_CreateTable.sql
 - 021_TransactionInsert_SP.sql
 - 022_VirtualCardGetByExternalReferenceId_SP.sql
 - 023_VirtualCard_VirtualCardStatus_Drop_Create.sql
 - 024_CardGetAllByAccountId_SP.sql

```
<EmbeddedResource Include="Migrations\180_LedgerAccount_Table_Update_SPs.sql" />
<EmbeddedResource Include="Migrations\181_Add_Column_ProductId_Ledger_Table.sql" />
<EmbeddedResource Include="Migrations\182_LedgerAccount_Table_Update_SPs_ProductId.sql" />
<EmbeddedResource Include="Migrations\183_LedgerAccount_Add_Index_External_Reference_Id.sql" />
<EmbeddedResource Include="Migrations\184_LedgerAccount_Add_Index_External_Ledger_Account_Id.sql" />
</ItemGroup>
```

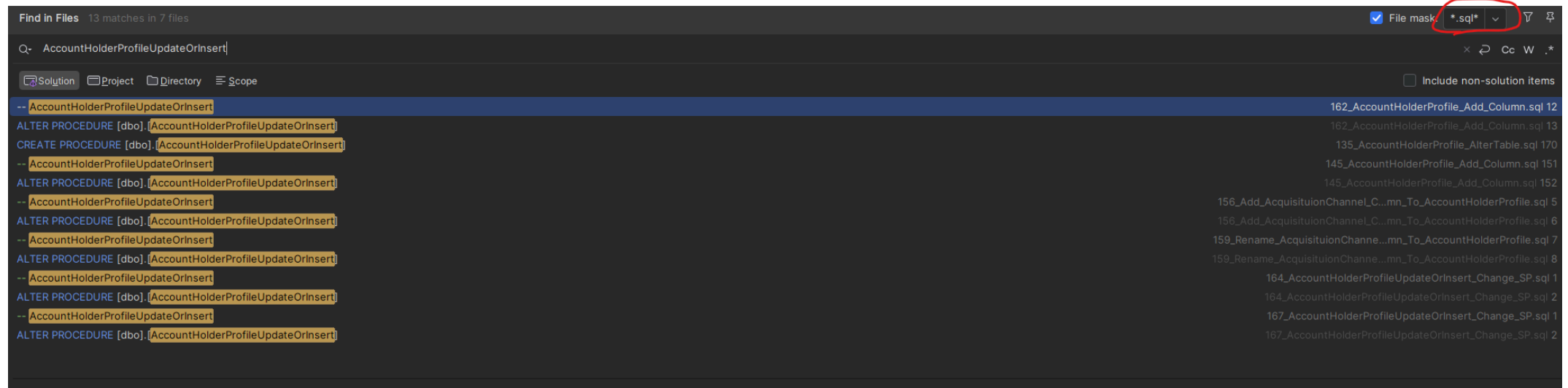
```
<ItemGroup>
  <EmbeddedResource Include="Migrations\*.sql" />
</ItemGroup>
```

[https://dev.azure.com/MAJORITY/Bank/_git/bank-wallet/pullrequest/47648?
_a=files&path=/Minority/Minority.Wallet/Minority.Wallet.Db/Migrations/182_LedgerAc
count_Table_Update_SPs_ProductId.sql](https://dev.azure.com/MAJORITY/Bank/_git/bank-wallet/pullrequest/47648?_a=files&path=/Minority/Minority.Wallet/Minority.Wallet.Db/Migrations/182_LedgerAccount_Table_Update_SPs_ProductId.sql)

Downsides

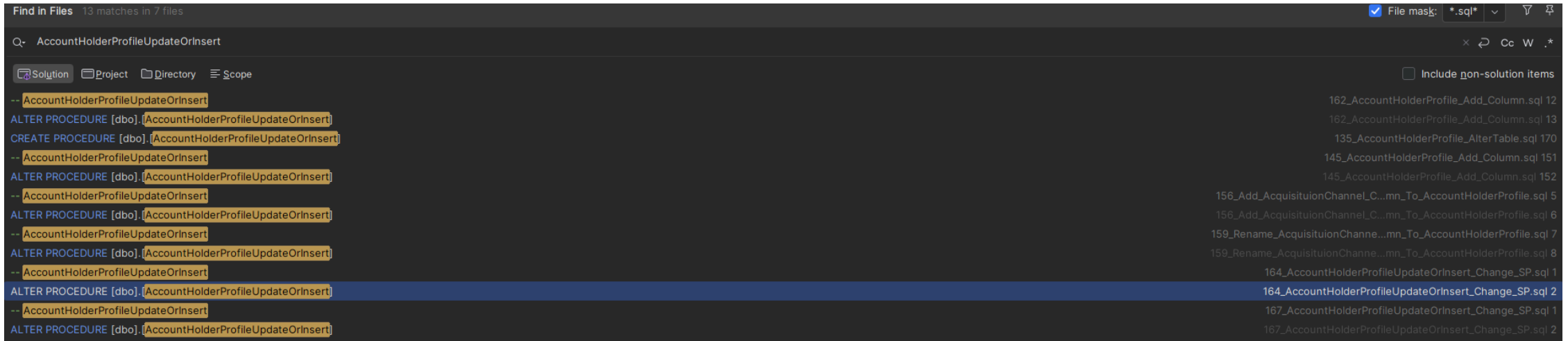
1. Lack of Version Clarity

When inspecting the code, it's challenging to determine the latest version of a stored procedure. This makes it time-consuming to understand the current production state.



2. Error-Prone Changes

When modifying a stored procedure, developers might mistakenly copy an outdated version, leading to potential production issues. (Including a risk to forget to do a git pull)



The screenshot shows an IDE's 'Find in Files' search results. The search query is 'AccountHolderProfileUpdateOrInsert'. The results are displayed in a list on the right side of the IDE, showing the file path and the line number where the search term was found. The search term is highlighted in yellow in the original image. The results are as follows:

File Path	Line Number
162_AccountHolderProfile_Add_Column.sql	12
162_AccountHolderProfile_Add_Column.sql	13
135_AccountHolderProfile_AlterTable.sql	170
145_AccountHolderProfile_Add_Column.sql	151
145_AccountHolderProfile_Add_Column.sql	152
156_Add_AcquisituonChannelC...mn_To_AccountHolderProfile.sql	5
156_Add_AcquisituonChannelC...mn_To_AccountHolderProfile.sql	6
159_Rename_AcquisituonChanne...mn_To_AccountHolderProfile.sql	7
159_Rename_AcquisituonChanne...mn_To_AccountHolderProfile.sql	8
164_AccountHolderProfileUpdateOrInsert_Change_SP.sql	1
164_AccountHolderProfileUpdateOrInsert_Change_SP.sql	2
167_AccountHolderProfileUpdateOrInsert_Change_SP.sql	1
167_AccountHolderProfileUpdateOrInsert_Change_SP.sql	2

3. Lack of Git Diffs

Since each modification requires copying the entire script to a new file, Git cannot provide meaningful diffs. Reviewing pull requests becomes harder as changes are not incremental.

[https://dev.azure.com/MAJORITY/Bank/_git/bank-wallet/pullrequest/47648?
path=/Minority/Minority.Wallet/Minority.Wallet.Db/Migrations/182_LedgerAccount
_Table_Update_SPs_ProductId.sql&_a=files](https://dev.azure.com/MAJORITY/Bank/_git/bank-wallet/pullrequest/47648?path=/Minority/Minority.Wallet/Minority.Wallet.Db/Migrations/182_LedgerAccount_Table_Update_SPs_ProductId.sql&_a=files)

4. Lack of Git Conflicts

Since each modification requires copying the entire script to a new file. Two developers modifying the same stored procedure in two different pull requests will not be guarded with a conflict when merged, which could lead to potential issues.

2. Improvement Suggestion: Custom Checksum Journaling

The Concept

- **Checksum Journal:** Implement a mechanism that allows tracking changes using a checksum instead of creating new files.
- **How It Works:** Instead of adding a new file for each change, update the existing script and use a checksum to identify when changes occur. This allows us to run scripts when their content changes. **"RunOnChange Scripts"**

3. Demo

4. Discussion

5. Summary (after discussion):

- No more creating "new" SQL files for altering stored procedures.
- Any area adopting this new approach should create a folder within /Migrations named StoredProcedures (/Migrations/StoredProcedures).
- Each script in the /Migrations/StoredProcedures folder should ideally contain only one stored procedure per file, making it easier to identify and apply changes independently.
- A validation mechanism will be introduced to minimize the risk of using the outdated method of adding new script files for stored procedures, This is to prevent copying the wrong sp which is no longer needed. This validation is only activated when a StoredProcedures folder is present, meaning no changes are enforced until the new approach is adopted by its area.
- While adopting this new approach, all SP:s needs to be moved into the StoredProcedures folder in same PR for the validation to work.