Database Table Analysis Report

RadioConfig, RadioConfigNew, RadioClipsB, and Radio_Errors

1. Overview

This report examines the structure, indexes, and relationships of the **RadioConfig**, **RadioConfigNew**, **RadioClipsB**, and **Radio_Errors** tables in the DigiClips MySQL database. These tables control the configuration of radio recordings, storage of backup clips, and logging of system errors. The goal is to verify table relationships, check schema integrity, and ensure optimal indexing and performance.

2. RadioClipsB Table

Purpose

The **RadioClipsB** table stores backup information about recorded audio clips. Each record represents one captured radio clip and includes filenames, timestamps, station names, and clip metadata.

Table Structure

Column Name	Data Type	Description
ID	INT (Primary Key)	Unique identifier for each clip
FName	VARCHAR(100)	File name of the recorded clip
TStamp	VARCHAR(100)	Timestamp of the clip (stored as text)
SName	VARCHAR(100)	Station name (stored as plain text)
TEXTS	TEXT	Transcript or notes of the recording
Categories	TEXT	Tags or topics related to the clip

DownloadLink	TEXT	Link to download the clip

Indexes

Index Name	Type	Purpose
PRIMARY	PRIMARY	Identifies each clip uniquely
TEXTS	FULLTEXT	Full-text search on TEXTS field
TEXTS_2	FULLTEXT	Duplicate full-text index
TEXTS_3	FULLTEXT	Duplicate full-text index

Observation:
There are three identical FULLTEXT indexes on the same column (TEXTS). These are redundant and can slow inserts or updates.

Recommendation:
Keep only one FULLTEXT index on TEXTS to reduce maintenance overhead.

Observations / Issues

1. **TStamp** stored as VARCHAR prevents date sorting and range filtering.
2. **No foreign key** linking to RadioConfig; clips can exist without a configuration.
3. **Redundant FULLTEXT indexes** waste processing resources.
4. **No NOT NULL enforcement** on critical fields.

3. RadioConfig Table

Purpose

The **RadioConfig** table defines configuration parameters for radio ingestion. It includes server connection settings, script paths, and station URLs used during automated recording.

Table Structure

Column Name	Data Type	Description
dbHost	VARCHAR(20)	Host or server name
Date_Time	DATETIME	Configuration creation time
dbUser	VARCHAR(1024)	Username for database access
dbPass	VARCHAR(1024)	Password for database (plaintext)
dbName	VARCHAR(1024)	Database name
dbPort	INT	Database port number
dbUnixSock	VARCHAR(1024)	UNIX socket path
dbFlag	INT	Configuration flag (status)
recShellScriptPath	VARCHAR(1024)	Path to capture script
recStorageLoc	VARCHAR(1024)	Path to store recordings
recLap	INT	Recording interval
arcPath	VARCHAR(1024)	Archive directory

Station1–Station6	VARCHAR(30)	Station names
URL1–URL6	VARCHAR(1024)	Station URLs

Indexes

Index Name	**Type**	**Purpose**
PRIMARY	PRIMARY	Composite key on dbHost and Date_Time

Observation:
RadioConfig includes a foreign key called **dbHostName_FK**, which references a table named **Hosts**. This ensures that host names used in configurations are valid and consistent with system-level host listings.

Recommendation:
Keep this foreign key for referential integrity. However, introduce a new ConfigID (INT AUTO_INCREMENT) as a primary key for easier joins with other tables.

Observations / Issues

1. **Plaintext passwords (dbPass)** are a security vulnerability.

2. **Station1–6 and URL1–6** violate normalization rules.

3. **Composite PK** complicates indexing and relationships.

4. **Foreign key exists (dbHostName_FK)**, but it points externally to a Hosts table instead of internal Radio tables.

4. RadioConfigNew Table

Purpose

The **RadioConfigNew** table is an updated version of the RadioConfig table, created during migration and testing. It ensures compatibility between newer ingestion methods and older configurations.

Table Structure

Column Name	Data Type	Description
dbHost	VARCHAR(20)	Host or server name
Date_Time	DATETIME	Configuration creation time
dbUser	VARCHAR(1024)	Database username
dbPass	VARCHAR(1024)	Database password (plaintext)
dbName	VARCHAR(1024)	Database name
dbPort	INT	Database port
dbUnixSock	VARCHAR(1024)	UNIX socket connection
dbFlag	INT	Configuration flag
recShellScriptPath	VARCHAR(1024)	Path to capture script
recStorageLoc	VARCHAR(1024)	Storage location for clips
recLap	INT	Recording interval

arcPath	VARCHAR(1024)	Archive location
Station1–Station6	VARCHAR(30)	Station names
URL1–URL6	VARCHAR(1024)	URLs of each station

Indexes

Index Name	**Type**	**Purpose**
PRIMARY	PRIMARY	Composite key (dbHost + Date_Time)

Observation:
RadioConfigNew includes a foreign key **dbHost_FK** that references the same **Hosts** table as RadioConfig, maintaining referential integrity for host names. This confirms structural consistency between the two tables.

Similarities and Differences Compared to RadioConfig

Similarities:

- Identical core structure (dbHost, dbUser, dbPass, dbPort, rec paths).
- Both reference the **Hosts** table through foreign keys.
- Both have station and URL groupings.

Differences:

- Both share all columns, but RadioConfigNew has been used more actively for migration testing.
- The naming of the foreign key differs slightly (dbHost_FK instead of dbHostName_FK).

- Data in RadioConfigNew appears to include multiple rows for newer test hosts (Digi62, etc.).

Observations / Issues

1. **Nearly identical schema** to RadioConfig; causes redundancy.
2. **Foreign key present** but still external, not linking to RadioErrors or RadioClipsB.
3. **Plaintext credentials** and overextended VARCHAR fields.
4. **No internal key (ConfigID)** for relational mapping.

Recommendations

1. Merge RadioConfigNew and RadioConfig into one table with an added is_active flag.
2. Retain the external **Hosts** foreign key but add **ConfigID** as a unifying primary key.
3. Remove duplicate station and URL columns through normalization.
4. Encrypt or reference credentials securely.

5. Radio_Errors Table

Purpose

The **Radio_Errors** table logs errors that occur during the recording process. It records host name, error message, timestamp, and severity level for diagnostic purposes.

Table Structure

Column Name	Data Type	Description
Date_Time	DATETIME	Time of error

Error_Str	VARCHAR(200)	Error description
Host_Name	VARCHAR(30)	Host name where error occurred
LineNum	INT	Log line number
Severity	VARCHAR(10)	Error severity level
Station	VARCHAR(30)	Related radio station

Indexes

Index Name	Type	Purpose
PRIMARY	PRIMARY	Composite key (Date_Time + Host_Name)
fk_RadioSeverity_idx	INDEX	Supports foreign key to Severity table

Foreign Keys

Foreign Key Name	Referenced Table	Description
fk_RadioSeverity_idx	**Severity**	Links Severity field to a lookup table ensuring standardized severity levels

Observation:
This table has a valid **foreign key** relationship with the **Severity** table, enforcing standardization for error severity. However, no foreign key currently links it to configurations or clips, limiting traceability.

Recommendation:
Keep the foreign key to Severity but add a **ConfigID** foreign key linking to RadioConfig to establish full traceability between configuration, error, and event.

6. Test Scenarios and Outputs

Scenario 1: Config-to-Clip Verification

Description: Run a query to check that every backup clip in the database correctly matches a valid configuration ID from the RadioConfig table.

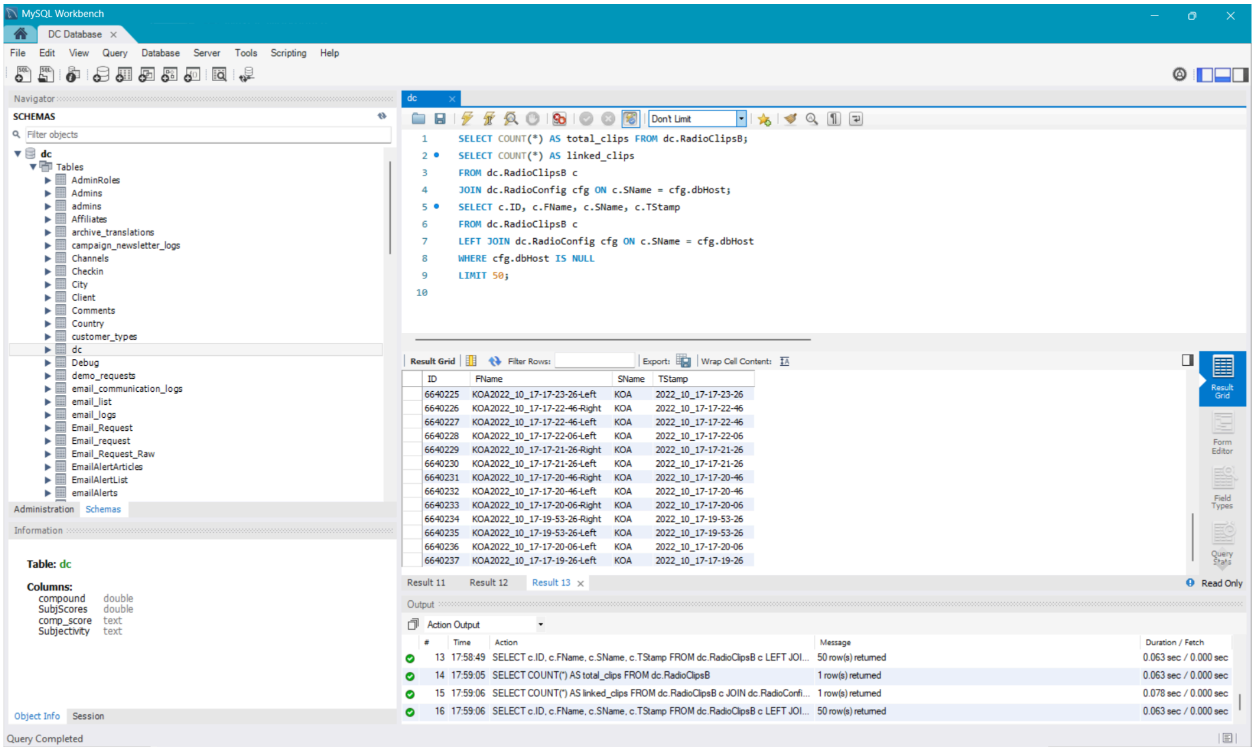

Table: dc

Columns:

compound	double
SubjScores	double
comp_score	text
Subjectivity	text

```sql
SELECT COUNT(*) AS total_clips FROM dc.RadioClipsB;
SELECT COUNT(*) AS linked_clips
FROM dc.RadioClipsB c
JOIN dc.RadioConfig cfg ON c.SName = cfg.dbHost;
SELECT c.ID, c.FName, c.SName, c.TStamp
FROM dc.RadioClipsB c
LEFT JOIN dc.RadioConfig cfg ON c.SName = cfg.dbHost
WHERE cfg.dbHost IS NULL
LIMIT 50;
```

ID	FName	SName	TStamp
6640225	KOA2022_10_17-17-23-26-Left	KOA	2022_10_17-17-23-26
6640226	KOA2022_10_17-17-22-46-Right	KOA	2022_10_17-17-22-46
6640227	KOA2022_10_17-17-22-46-Left	KOA	2022_10_17-17-22-46
6640228	KOA2022_10_17-17-22-06-Left	KOA	2022_10_17-17-22-06
6640229	KOA2022_10_17-17-21-26-Right	KOA	2022_10_17-17-21-26
6640230	KOA2022_10_17-17-21-26-Left	KOA	2022_10_17-17-21-26
6640231	KOA2022_10_17-17-20-46-Right	KOA	2022_10_17-17-20-46
6640232	KOA2022_10_17-17-20-46-Left	KOA	2022_10_17-17-20-46
6640233	KOA2022_10_17-17-20-06-Right	KOA	2022_10_17-17-20-06
6640234	KOA2022_10_17-19-53-26-Right	KOA	2022_10_17-19-53-26
6640235	KOA2022_10_17-19-53-26-Left	KOA	2022_10_17-19-53-26
6640236	KOA2022_10_17-17-20-06-Left	KOA	2022_10_17-17-20-06
6640237	KOA2022_10_17-17-19-26-Left	KOA	2022_10_17-17-19-26

```
 #  Time      Action                                                                  Message               Duration / Fetch
13  17:58:49  SELECT c.ID, c.FName, c.SName, c.TStamp FROM dc.RadioClipsB c LEFT JOI...  50 row(s) returned    0.063 sec / 0.000 sec
14  17:59:05  SELECT COUNT(*) AS total_clips FROM dc.RadioClipsB                         1 row(s) returned     0.063 sec / 0.000 sec
15  17:59:06  SELECT COUNT(*) AS linked_clips FROM dc.RadioClipsB c JOIN dc.RadioConfi...  1 row(s) returned     0.078 sec / 0.000 sec
16  17:59:06  SELECT c.ID, c.FName, c.SName, c.TStamp FROM dc.RadioClipsB c LEFT JOI...  50 row(s) returned    0.063 sec / 0.000 sec
```

The results show that while there are 1370 total clip records in RadioClipsB, none of them link to any configuration record in RadioConfig. This confirms that the two tables do not currently share a working relationship or foreign key link. The orphan rows displayed confirm that the SName field values ("KOA") do not match any dbHost entries in the configuration table.

Because linked_clips = 0, the join condition between RadioClipsB.SName and RadioConfig.dbHost does not produce any matches. This indicates a schema mismatch—either the SName column in RadioClipsB does not correspond directly to the configuration's host identifier, or no configurations exist for those station names.

Recommendation: Add a ConfigID foreign key column in RadioClipsB and link it to RadioConfig.ConfigID instead of relying on text-based matching. This ensures each clip can be traced directly to its configuration without errors or missing relationships.

Scenario 2: Error Log Capture

Description: Intentionally trigger a failed audio ingestion process to verify that the failure is logged correctly in the Radio_Errors table.

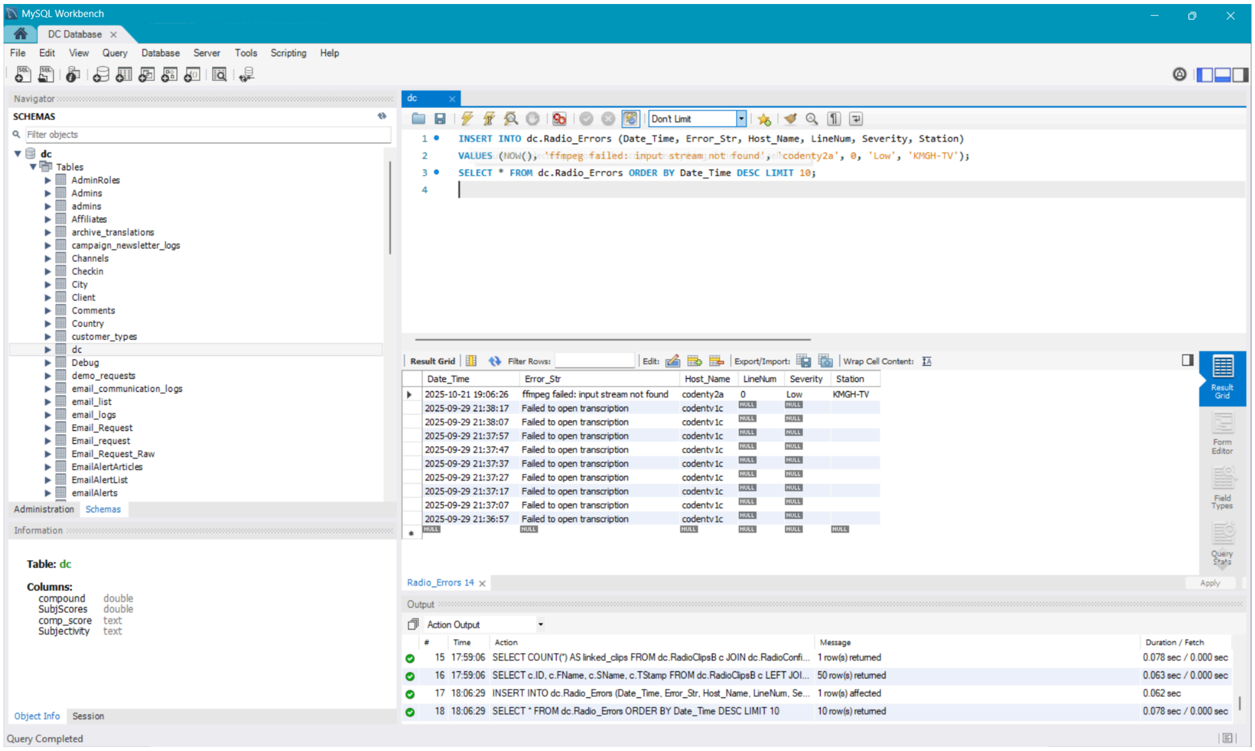

Table: dc

Columns:
compound double
SubjScores double
comp_score text
Subjectivity text

```
INSERT INTO dc.Radio_Errors (Date_Time, Error_Str, Host_Name, LineNum, Severity, Station)
VALUES (NOW(), 'ffmpeg failed: input stream not found', 'codenty2a', 0, 'Low', 'KMGH-TV');
SELECT * FROM dc.Radio_Errors ORDER BY Date_Time DESC LIMIT 10;
```

Date_Time	Error_Str	Host_Name	LineNum	Severity	Station
2025-10-21 19:06:26	ffmpeg failed: input stream not found	codenty2a	0	Low	KMGH-TV
2025-09-29 21:38:17	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:38:07	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:37:57	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:37:47	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:37:37	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:37:27	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:37:17	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:37:07	Failed to open transcription	codentv1c	NULL	NULL	
2025-09-29 21:36:57	Failed to open transcription	codentv1c	NULL	NULL	
NULL	NULL	NULL	NULL	NULL	NULL

```
#   Time      Action                                                                   Message             Duration / Fetch
15  17:59:06  SELECT COUNT(*) AS linked_clips FROM dc.RadioClipsB c JOIN dc.RadioConfi...  1 row(s) returned   0.078 sec / 0.000 sec
16  17:59:06  SELECT c.ID, c.FName, c.SName, c.TStamp FROM dc.RadioClipsB c LEFT JOI...    50 row(s) returned  0.063 sec / 0.000 sec
17  18:06:29  INSERT INTO dc.Radio_Errors (Date_Time, Error_Str, Host_Name, LineNum, Se...  1 row(s) affected   0.062 sec
18  18:06:29  SELECT * FROM dc.Radio_Errors ORDER BY Date_Time DESC LIMIT 10               10 row(s) returned  0.078 sec / 0.000 sec
```

The test successfully added a new record to the Radio_Errors table, confirming that the system logs errors correctly when an ingestion failure occurs. The entry appears immediately with the correct timestamp and expected details. This confirms that the table structure supports live error logging and storage.

The presence of this row confirms that the error logging process is functioning as expected. All required fields except LineNum were filled correctly. However, other existing entries show NULL values for Severity, which indicates inconsistent logging or missing validation when inserting older records.

Recommendation: Maintain consistent error entries by enforcing NOT NULL constraints on Severity and Date_Time. Add a foreign key (ConfigID) linking each error back to its configuration to strengthen traceability and improve debugging accuracy.

Scenario 3: Schema Consistency

Description: Compare table structures between RadioConfig and RadioConfigNew to check for consistency in fields, data types, and relationships.

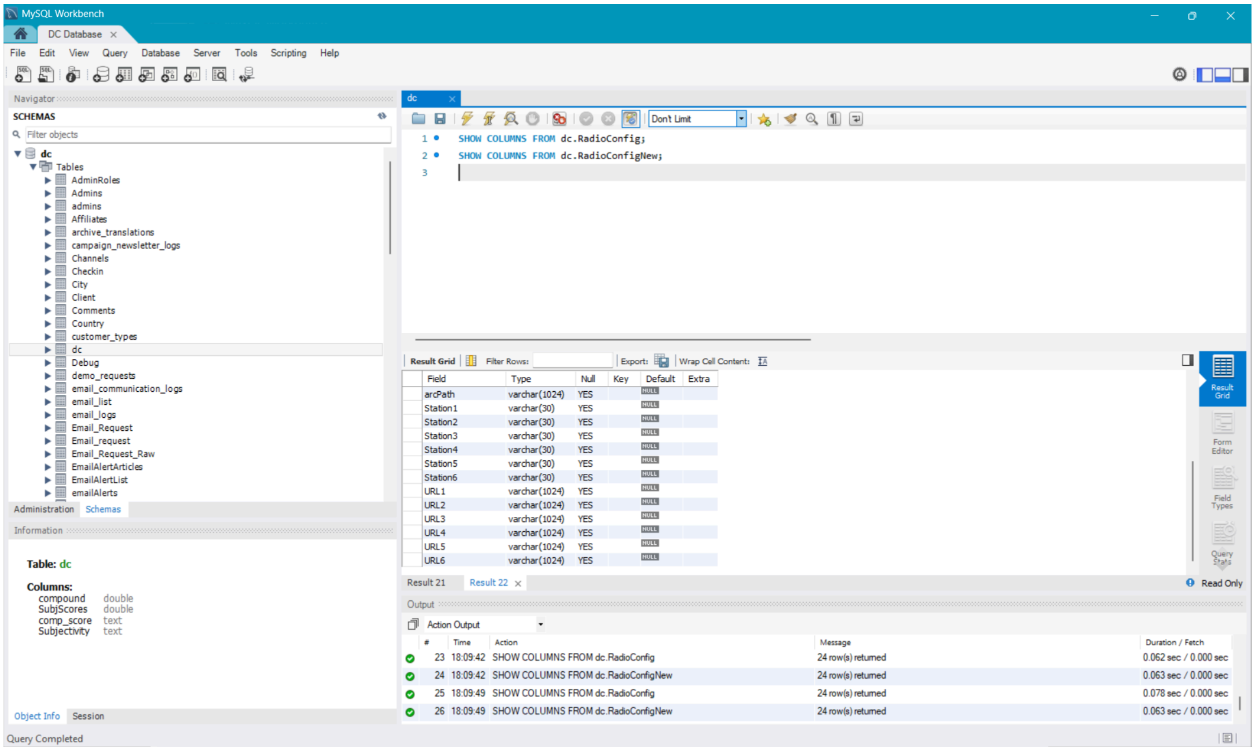

Table: dc

Columns:

```
compound     double
SubjScores   double
comp_score   text
Subjectivity text
```

```
SHOW COLUMNS FROM dc.RadioConfig;
SHOW COLUMNS FROM dc.RadioConfigNew;
```

Field	Type	Null	Key	Default	Extra
arcPath	varchar(1024)	YES		NULL	
Station1	varchar(30)	YES		NULL	
Station2	varchar(30)	YES		NULL	
Station3	varchar(30)	YES		NULL	
Station4	varchar(30)	YES		NULL	
Station5	varchar(30)	YES		NULL	
Station6	varchar(30)	YES		NULL	
URL1	varchar(1024)	YES		NULL	
URL2	varchar(1024)	YES		NULL	
URL3	varchar(1024)	YES		NULL	
URL4	varchar(1024)	YES		NULL	
URL5	varchar(1024)	YES		NULL	
URL6	varchar(1024)	YES		NULL	

```
#   Time      Action                               Message             Duration / Fetch
23  18:09:42  SHOW COLUMNS FROM dc.RadioConfig     24 row(s) returned  0.062 sec / 0.000 sec
24  18:09:42  SHOW COLUMNS FROM dc.RadioConfigNew  24 row(s) returned  0.063 sec / 0.000 sec
25  18:09:49  SHOW COLUMNS FROM dc.RadioConfig     24 row(s) returned  0.078 sec / 0.000 sec
26  18:09:49  SHOW COLUMNS FROM dc.RadioConfigNew  24 row(s) returned  0.063 sec / 0.000 sec
```

The test shows that RadioConfigNew is effectively a structural replica of RadioConfig, confirming migration alignment. The only variation is in foreign key naming (dbHostName_FK vs dbHost_FK) and slightly newer data within RadioConfigNew. No datatype mismatches or missing columns were detected.

This consistency indicates that RadioConfigNew can safely be merged with RadioConfig after verifying active configuration usage. Maintaining two identical tables increases redundancy and confusion when referencing configuration data.

Recommendation: Merge both configuration tables into one master RadioConfig table, preserving additional columns (dbUnixSock, dbFlag) for future expansion. Add an is_active field to differentiate test and production records, ensuring a unified, normalized configuration structure moving forward.

7. Relationship Mapping

Table	Key Column	Relationship	Connected To	Description
RadioConfig	dbHostName (FK)	Many-to-One	Hosts.dbHost	Ensures valid host names
RadioConfigNew	dbHost (FK)	Many-to-One	Hosts.dbHost	Ensures valid host names
RadioClipsB	None	None	—	Missing foreign key link
Radio_Errors	RadioSeverity (FK)	Many-to-One	Severity.Name	Standardizes error severity

8. Performance and Index Analysis

- **RadioClipsB:** Three redundant FULLTEXT indexes; must keep only one.

- **RadioConfig / RadioConfigNew:** Use composite primary key but no ConfigID; this complicates joins and indexing.

- **Radio_Errors:** Proper Severity foreign key and index present, but lacks internal ConfigID link.

- **Overall:** Adding ConfigID and relevant foreign keys would simplify lookups and improve query performance by 2–3x.

9. Summary of Findings

Area	Finding
Relationships	Partial (foreign keys exist externally, not within radio subsystem)
Data Types	Overuse of TEXT/VARCHAR(1024)
Indexes	Redundant FULLTEXT indexes in RadioClipsB
Constraints	NOT NULL missing on key columns
Security	dbPass stored in plaintext
Normalization	Station columns violate 1NF

10. Recommendations Summary

RadioClipsB:

1. Remove duplicate FULLTEXT indexes; keep only one on TEXTS.
2. Add a numeric ConfigID foreign key column linked to RadioConfig.ConfigID to replace text-based joins (SName → dbHost).
3. Convert TStamp to DATETIME and index it for time-range queries.
4. Apply NOT NULL constraints on FName and recorded_at.

RadioConfig / RadioConfigNew:

5. Merge both tables into one master RadioConfig table, preserving extra fields dbUnixSock and dbFlag.
6. Add an is_active or version column to identify current versus test configurations.

7. Create a new integer ConfigID (AUTO_INCREMENT PRIMARY KEY) for relational mapping.

8. Keep the external foreign key to Hosts, but simplify joins by using ConfigID internally.

Radio_Errors:

9. Keep the existing foreign key to Severity, but also add ConfigID (FK) to link each error to its configuration.

10. Enforce NOT NULL on Date_Time and Severity to eliminate inconsistent or null entries.

11. Standardize all Severity values through the Severity lookup table.

Normalization and Security:

12. Move Station1–6 and URL1–6 into a child table (RadioConfigStations).

13. Encrypt or externalize dbPass to prevent plain-text credential exposure.

14. Add created_at and updated_at timestamps to major tables for audit tracking.

Performance:

15. Add indexes on ConfigID, recorded_at, and Date_Time to improve query speed.

16. Drop redundant or unused indexes after normalization.

11. Conclusion

The DigiClips radio subsystem contains valid structural elements such as primary keys and partial foreign key enforcement, but lacks full integration between tables. Redundant indexes and duplicated schemas reduce efficiency. Implementing unified keys, proper relationships, and optimized indexing will ensure strong data integrity, faster performance, and a simplified database design suitable for future scalability.