HIRELINK

Database Design Document

Architecture, Normalization & Optimization

Version 1.0 | January 2026

Table of Contents

Table of Contents 2
1. Executive Summary 3
1.1 Design Goals 3
1.2 Technology Stack 3
2. Database Architecture 3
2.1 Schema Overview 3
2.2 Table Statistics 3
3. Normalization Analysis 4
3.1 First Normal Form (1NF) 4
3.2 Second Normal Form (2NF) 4
3.3 Third Normal Form (3NF) 4
3.4 Strategic Denormalization 4
4. Indexing Strategy 4
4.1 Primary Key Indexes 4
4.2 Secondary Indexes 4
4.3 Composite Indexes 5
5. Data Types Selection 5
5.1 Numeric Types 5
5.2 String Types 5
5.3 Date/Time Types 5
6. Security Considerations 6
6.1 Sensitive Data Protection 6
6.2 Access Control 6
6.3 Audit Trail 6
7. Performance Optimization 6
7.1 Query Optimization Techniques 6
7.2 Caching Strategy 6
7.3 Partitioning Strategy (Future) 7
8. Backup and Recovery 7
8.1 Backup Strategy 7
8.2 Recovery Procedures 7
9. Revision History 7

1. Executive Summary

This document outlines the database design for HireLink, a local service provider platform. The database is designed to be scalable, performant, and maintainable while supporting the complex business requirements of connecting service providers with customers.

1.1 Design Goals

- Scalability: Support millions of users and transactions
- Performance: Sub-second response times for common queries
- Data Integrity: Enforce referential integrity and business rules
- Security: Protect sensitive user and payment data
- Maintainability: Clear structure for easy updates and debugging
- Flexibility: Support future feature additions without major rewrites

1.2 Technology Stack

Component	Technology
Database Engine	MySQL 8.0+ with InnoDB storage engine
Character Set	utf8mb4 (full Unicode support including emojis)
Collation	utf8mb4_unicode_ci (case-insensitive comparison)
Connection Pool	HikariCP for Spring Boot backend

2. Database Architecture

2.1 Schema Overview

The HireLink database consists of 18 tables organized into logical modules:

Module	Tables	Purpose
User Management	4	User accounts, sessions, OTP, addresses
Provider Management	4	Provider profiles, documents, availability, areas
Service Catalog	2	Categories and services
Booking System	2	Bookings and status history
Payment System	2	Payments and transactions
Communication	3	Conversations, messages, notifications
Reviews	1	User reviews and ratings

2.2 Table Statistics

Table	Columns	Indexes	Expected Rows (Year 1)
users	20	5	100,000+
service_providers	26	7	10,000+
services	17	5	50,000+
bookings	26	8	500,000+
payments	19	6	500,000+
reviews	18	6	200,000+
messages	13	4	2,000,000+
notifications	12	4	5,000,000+

3. Normalization Analysis

The HireLink database follows normalization principles up to Third Normal Form (3NF) with strategic denormalization for performance optimization.

3.1 First Normal Form (1NF)

All tables satisfy 1NF requirements:

- Each column contains only atomic (indivisible) values
- Each row is uniquely identifiable by a primary key
- No repeating groups or arrays stored directly
- JSON columns used only for flexible schema data (preferences, images array)

3.2 Second Normal Form (2NF)

All tables satisfy 2NF requirements:

- All non-key attributes fully depend on the entire primary key
- No partial dependencies in composite keys
- Junction tables properly decomposed

3.3 Third Normal Form (3NF)

All tables satisfy 3NF requirements:

- No transitive dependencies between non-key attributes
- All non-key attributes depend only on the primary key
- Lookup data separated into reference tables

3.4 Strategic Denormalization

Certain fields are intentionally denormalized for performance:

Table	Denormalized Field	Justification
service_providers	average_rating	Avoid calculating average on every query; updated via trigger
service_providers	total_bookings	Counter cache for quick stats display
service_providers	completion_rate	Pre-calculated percentage for sorting/filtering
conversations	last_message_text	Preview text without joining messages table
conversations	unread_count	Quick badge display without counting
bookings	user_rating	Quick access without joining reviews
services	times_booked	Popularity ranking without count query

4. Indexing Strategy

4.1 Primary Key Indexes

All tables use BIGINT UNSIGNED AUTO_INCREMENT primary keys. InnoDB automatically creates clustered indexes on primary keys.

4.2 Secondary Indexes

Table	Index Columns	Use Case
users	phone	Login lookup by phone number
users	email	Login lookup by email
users	user_type	Filter users by type
service_providers	base_latitude, base_longitude	Geospatial queries for nearby providers
service_providers	base_pincode	Filter by service area
service_providers	average_rating DESC	Sort by rating
service_providers	kyc_status	Admin verification queue
services	provider_id	List provider's services
services	category_id	Browse by category
bookings	user_id	User's booking history
bookings	provider_id	Provider's bookings
bookings	booking_status	Filter by status
bookings	scheduled_date, scheduled_time	Calendar queries
payments	gateway_payment_id	Webhook reconciliation
messages	conversation_id, is_read	Unread message count

4.3 Composite Indexes

Composite indexes are used for frequently combined query conditions:

- (provider_id, pincode) on provider_service_areas - unique constraint
- (provider_id, day_of_week) on provider_availability - schedule lookup
- (user_id, is_read) on notifications - unread notifications query
- (conversation_id, sent_at) on messages - message pagination

5. Data Types Selection

5.1 Numeric Types

Type	Usage	Rationale
BIGINT UNSIGNED	Primary/Foreign Keys	Supports up to 18 quintillion records; prevents negative IDs
INT UNSIGNED	Counters, years	Sufficient range for counts; smaller storage
DECIMAL(10,2)	Prices	Exact decimal for monetary values up to 99,999,999.99
DECIMAL(12,2)	Large amounts	Total earnings, payments up to billions
DECIMAL(3,2)	Ratings	Rating values 0.00 to 5.00
DECIMAL(10,8)	Latitude	8 decimal places for ~1mm precision
DECIMAL(11,8)	Longitude	Extra digit for values up to ±180

5.2 String Types

Type	Usage	Rationale
VARCHAR(6)	Pincode	Indian pincodes are exactly 6 digits
VARCHAR(15)	Phone	International phone with country code
VARCHAR(100)	Names	Standard name length limit
VARCHAR(150)	Email	RFC 5321 allows up to 254 chars
VARCHAR(255)	Short text	Single-byte length prefix optimization
VARCHAR(500)	URLs, encrypted	URL-safe buffer; encrypted data length
TEXT	Descriptions	Variable length up to 65KB
JSON	Arrays, objects	Native JSON support with indexing

5.3 Date/Time Types

Type	Usage	Rationale
TIMESTAMP	created_at, updated_at	Auto timezone conversion; range 1970-2038
DATE	scheduled_date, dob	Date only without time component
TIME	scheduled_time	Time only for scheduling

6. Security Considerations

6.1 Sensitive Data Protection

Data Type	Protection Method	Implementation
Passwords	BCrypt Hashing	Cost factor 12; stored as password_hash
Aadhaar Numbers	AES-256 Encryption	Encrypted at application layer
PAN Numbers	AES-256 Encryption	Encrypted at application layer
JWT Tokens	SHA-256 Hashing	Only hash stored; token not recoverable
OTP Codes	SHA-256 Hashing	Time-limited; hash comparison only

6.2 Access Control

- Database user with minimal required privileges for application
- Separate read-only user for reporting queries
- Admin operations through separate high-privilege account
- Connection encryption via TLS 1.3
- IP whitelist for database access

6.3 Audit Trail

The admin_audit_logs table captures all administrative actions with:

- Admin user identification
- Action type and description
- Target entity and ID
- Before and after values (JSON)
- IP address and timestamp

7. Performance Optimization

7.1 Query Optimization Techniques

- Use covering indexes for frequently accessed columns
- Avoid SELECT * - specify only needed columns
- Use EXPLAIN ANALYZE to verify query plans
- Batch inserts for bulk operations
- Connection pooling with HikariCP

7.2 Caching Strategy

Data	Cache Layer	TTL
Service Categories	Redis	24 hours (infrequently changed)
Provider Profiles	Redis	1 hour (moderate changes)

Data	Cache Layer	TTL
User Sessions	Redis	Session duration
Search Results	Redis	5 minutes (real-time needed)
System Settings	Application	On startup / manual refresh

7.3 Partitioning Strategy (Future)

For tables expected to exceed 100 million rows:

- bookings: Partition by RANGE on scheduled_date (monthly)
- messages: Partition by RANGE on sent_at (monthly)
- notifications: Partition by RANGE on created_at (weekly)
- payment_transactions: Partition by RANGE on created_at (monthly)

8. Backup and Recovery

8.1 Backup Strategy

Type	Frequency	Retention	Method
Full Backup	Daily	30 days	mysqldump / AWS RDS
Incremental	Hourly	7 days	Binary logs
Point-in-time	Continuous	7 days	RDS PITR
Cross-region	Daily	90 days	S3 replication

8.2 Recovery Procedures

- RTO (Recovery Time Objective): 1 hour for critical systems
- RPO (Recovery Point Objective): 5 minutes maximum data loss
- Automated failover to read replica in case of primary failure
- Regular recovery drills to validate backup integrity

9. Revision History

Version	Date	Author	Changes
1.0	January 2026	HireLink Team	Initial design