SMART PARKING MANAGEMENT SYSTEM

PL/SQL Practicum Project Description

Student Name: Sugira Ghislain **ID**:27776| **Category:** Business/Smart City | **Date:** November 14, 2025

---

1. PROJECT OVERVIEW

Problem: Urban parking causes 30% of city traffic, wasting time and fuel. Traditional systems lack real-time tracking and automated billing.

Solution: A database-driven Smart Parking Management System automating slot tracking, dynamic pricing, billing, and reporting for parking facilities, malls, and commercial complexes.

Key Features: Real-time availability, automated entry/exit, dynamic pricing, reservation system, payment processing, analytics/reporting, and overstay alerts.

2. DATABASE SCHEMA (7 Tables)

PARKING_LOTS lot_id (PK), lot_name, location total_slots, status, created_date

PARKING_SLOTS slot_id (PK), lot_id (FK) slot_number, slot_type, floor_level status, hourly_rate

VEHICLES vehicle_id (PK), plate_number (UQ) vehicle_type, owner_name owner_contact, is_frequent

PARKING_SESSIONS session_id (PK), slot_id (FK) vehicle_id (FK), entry_time exit_time, session_status parking_fee

PAYMENTS payment_id (PK), session_id (FK) amount, payment_method payment_date, transaction_ref discount_applied

RESERVATIONS reservation_id (PK), vehicle_id (FK) slot_id (FK), reservation_time reserved_from, reserved_until status

AUDIT_LOG log_id (PK), table_name operation_type, record_id old_value, new_value changed_by, change_date

3. PL/SQL COMPONENTS

Procedures (5)

- park_vehicle() - Process entry
- exit_vehicle() - Calculate fee & exit
- reserve_slot() - Advance booking
- cancel_reservation() - Free slots
- generate_daily_report() - Analytics

Triggers (5)

- TRG_UPDATE_SLOT_STATUS - Auto-update
- TRG_VALIDATE_RESERVATION - Prevent double-booking
- TRG_AUDIT_PAYMENTS - Log transactions
- TRG_EXPIRE_RESERVATIONS - Auto-expire

Functions (5)

- calculate_parking_fee() - Fee calculation
- get_available_slots() - Count free slots
- check_slot_availability() - Validation
- get_monthly_revenue() - Revenue sum
- calculate_occupancy_rate() - Usage %
- TRG_CHECK_SLOT_AVAILABLE - Validate

Packages (3)

- PKG_PARKING_OPERATIONS - Core ops
- PKG_PAYMENT_PROCESSING - Payments
- PKG_REPORTS - Analytics

Cursors

Active sessions listing, overstay vehicle identification, bulk monthly billing and report generation.

4. INNOVATION & IMPROVEMENTS

① **Dynamic Pricing Engine:** Time-of-day pricing (peak hours 1.5x rate, off-peak standard, overnight discounted) and demand-based rates replace traditional fixed pricing.

② **Automated Loyalty Program:** Frequent parkers (10+ visits/month) receive automatic 15% discounts. Tier-based benefits unlock with accumulated hours—no manual intervention required.

③ **Predictive Slot Recommendation:** Algorithm analyzes historical data to suggest nearest available slots, predict vacancy patterns, and recommend optimal parking duration.

④ **Smart Overstay Management:** Automated alerts when parking exceeds duration, 15-minute grace period, escalating penalty fees for extended overstays.

⑤ **Real-time Analytics Dashboard:** Pre-calculated views showing occupancy percentage, revenue trends, peak hours, and average duration metrics.

⑥ **Complete Audit Trail:** Every database modification logged for dispute resolution, tampering detection, and regulatory compliance.

5. EXPECTED OUTCOMES

Business Benefits

- 40% faster entry/exit processing
- 25% revenue increase (dynamic pricing)
- 60% less manual intervention
- Transparent billing system

Technical Achievements

- Advanced PL/SQL implementation
- Complex business logic & validation
- Normalized database design
- Transaction & error handling

6. TECHNICAL IMPLEMENTATION HIGHLIGHTS

Data Integrity: Foreign key constraints ensure referential integrity. Check constraints validate slot types (REGULAR/VIP/HANDICAP/MOTORCYCLE) and payment methods (CASH/CARD/MOBILE/WALLET).

Automation: Triggers automatically update slot status on vehicle entry/exit, expire reservations past their time window, and maintain audit logs without manual intervention.

Performance Optimization: Indexes on frequently queried columns (plate_number, slot_number, entry_time). Bulk operations for monthly processing reduce execution time.

Error Handling: Exception blocks in all procedures handle duplicate entries, invalid slots, payment failures, and concurrent reservation conflicts.

7. CONCLUSION

The Smart Parking Management System solves real-world urban challenges through intelligent database architecture and automated PL/SQL programming. Combining dynamic pricing, predictive analytics, and comprehensive audit trails, this project demonstrates technical proficiency and innovative problem-solving. The system is extensible for future integration with mobile apps, IoT sensors, and AI-based demand forecasting.