Phase 4 – Analytical Query Development & BI Modeling

Overview

Phase 4 focused on developing, validating, and documenting **analytical SQL queries** that power the business-intelligence layer of the Airline BI Database.
Where Phase 3 ensured the **data was clean, constrained, and trustworthy**, Phase 4 leveraged that foundation to produce a portfolio of **15 production-ready analytical queries** across four categories:

- CTE-based reporting queries
- Window-function analytical models
- Recursive network-analysis queries
- Complex joins & aggregations for operational insights

All work was performed in:

- `notebooks/03_analytics_queries.ipynb` — query development, testing, and result validation
- `docs/phase_4_query_catalog.md` — final curated catalog of business questions, SQL, and outputs
- `docs/phase_4_analytics.png` — summary visualizations (flight status, revenue mix, delays)

This phase enables the BI, analytics, and dashboard layers used in Phase 5.

---

1. Analytical Query Framework

The Phase 4 query framework was designed to support:

- operational reporting (delays, flight volumes)
- commercial analytics (revenue by fare class, CLV)
- loyalty segmentation (tier transitions, top-value members)
- network modeling (connectivity, multi-hop routes)
- payment funnel performance (success rates by channel)

Each query was written to be:

- deterministic and re-runnable
- aligned with schema constraints from Phase 3
- compatible with SQLAlchemy / PostgreSQL
- optimized through indexing and CTE inlining
- validated with sample outputs in the development notebook

The result is a complete analytics layer suitable for dashboards, materialized views, or API endpoints in a production BI system.

---

2. CTE-Based Reporting Queries

Five core analytical questions were expressed using CTEs for clarity and modularity:

1. **Top 10 busiest airports**
   Combined arrivals + departures to identify system bottlenecks and high-traffic hubs.

2. **Airline on-time performance (BTS)**
   Summarized massive BTS performance data into airline-level KPIs (delays, cancellations, diversions).

3. **Monthly passenger counts**
   Aggregated bookings into calendar months to track demand and seasonality.

4. **Loyalty tier transitions**
   Compared current tier vs. miles-based target tier to reveal upgrade/downgrade candidates.

5. **Revenue per fare class**
   Combined bookings + payments to compute revenue mix and average yield per fare product.

These CTEs provide the foundation for common business questions in airline commercial and operations analytics.

---

3. Window-Function Analytical Models

Five queries leveraged window functions to model trends, rankings, percentiles, and cumulative metrics:

1. **Airline ranking by average delay**
   `RANK()` used to identify the worst operational performers.

2. **Running monthly revenue totals**
   `SUM() OVER(ORDER BY month)` produced cumulative revenue pacing.

3. **Percent of flights delayed by month**
   `AVG(CASE WHEN ...)` paired with month grouping for reliability reporting.

4. **Customer Lifetime Value (CLV)**
   Passenger-level `SUM(amount) OVER(PARTITION BY passenger ORDER BY date)` created a rolling lifetime value model.

5. **Dense_rank route distance analysis**
   `DENSE_RANK()` applied to great-circle distances to rank longest routes.

These models support financial reporting, revenue forecasting, loyalty targeting, and network planning.

---

4. Recursive CTE Network Analysis

Two recursive queries provide high-value network-planning insights:

1. **Airport connectivity graph (from busiest origin)**
   Generated full reachable airport sets within 1–3 hops, revealing the true reach of a hub.

2. **Multi-hop route expansion (up to 3 hops)**
   Enumerated all path combinations with their hop counts, supporting connection quality analysis and identifying potential opportunities for direct service.

Both recursive models rely entirely on the cleaned `routes` table built in Phase 3 and backfilled in early Phase 4.

---

5. Complex Joins & Aggregations

Three additional advanced queries provided deeper operational and commercial insights:

1. **Payment success rate by booking channel**
   Quantified funnel efficiency for Web, Mobile, Call Center, and Agents.

2. **Worst routes by delay + cancellations**
   Combined delay averages and cancellation ratios to surface operational problem routes.

3. **High-value loyalty members (top 5%)**
   Used `CUME_DIST()` to extract the most valuable segment of the loyalty base.

These aggregations directly support revenue management, retention marketing, and operations management.

---

6. Performance Testing & Optimization

Each of the 15 analytical queries was validated using:

- **EXPLAIN** — plan inspection
- **EXPLAIN ANALYZE** — timing and row-level execution review
- **Index verification** — ensured all joins used Phase 3 indexes

- **CTE inlining checks** — confirmed PostgreSQL inlined or executed efficiently
- **Join cardinality validation** — ensured expected row counts

Key observations:

- Indexes created in Phase 3 (especially on `flights(flight_date)` and `payments(paid_at)`) materially reduced scan times.
- CTEs were automatically inlined by PostgreSQL 12+, avoiding unnecessary materialization.
- Recursive CTE depth was capped at 3–4 hops to prevent runaway combinatorics.
- No analytical queries required additional indexes or materialized views, though several will be migrated to MVs in Phase 5.

---

7. Visual Analytics Artifacts

Three visual proofs were generated in `03_analytics_queries.ipynb` and exported to:

- `docs/phase_4_analytics.png`

The visuals include:

1. **Flight Status Distribution**
   Bar chart showing operational health of the synthetic network.
2. **Revenue by Fare Class**
   Clear revenue mix and yield comparison across fare families.
3. **Delay Distribution Histogram**
   Shows skewness and outliers in delay minutes.

These charts validate the underlying data and demonstrate BI-readiness of the analytical layer.

---

8. Deliverables Produced in Phase 4

- `notebooks/03_analytics_queries.ipynb`
  Full query development notebook with outputs and validation.

- `docs/phase_4_query_catalog.md`
  Canonical documentation of all 15 business questions, SQL, inputs/outputs, and interpretation.

- `docs/phase_4_analytics.png`
  Visual proof set used for dashboard design.

- `sql/phase_4_perf_tests.sql`
  EXPLAIN/ANALYZE performance validations.

- `etl/backfill_routes_aircraft_changes.py` (used earlier in Phase 4)
  Ensured analytical tables (routes, aircraft) were populated for network queries.

These artifacts complete the analytical layer needed for Phase 5's BI dashboards and materialized view design.

---

9. Phase 4 Summary

Phase 4 successfully transformed the cleaned, standardized database into a **fully analytical BI engine**.
The 15 SQL models now form the backbone of:

- operational dashboards
- commercial performance analytics
- loyalty program insights
- network planning and forecasting
- revenue and payment funnel analysis

With the analytical layer complete, the project is ready to proceed to **Phase 5**, where these queries are operationalized as:

- BI dashboards
- scheduled materialized views
- reporting endpoints
- executive-level analytics summaries

Phase 4 concludes with a robust, performant, and fully documented analytical query portfolio.