UNDERSTANDING WORKING ON SNOWFLAKE UNISTORE ZERO TO HERO

BASIC DISCUSSION

1. What is a Transactional (OLTP) workload?

🔍 **Explanation:** Transactional systems are optimized for **frequent, short, real-time operations** like inserts, updates, and deletes.

🏦 **FinTech Use Case:**In **Revolut**, each user tap (adding money, transferring funds) is a transaction that must be processed in **milliseconds** — no delay allowed.

🛠️ **Example (OLTP-style query):**

```
-- Inserting a payment transaction
INSERT INTO transactions (transaction_id, user_id, amount, type, timestamp)
VALUES ('txn_1001', 'usr_001', 250.00, 'DEBIT', CURRENT_TIMESTAMP);
```

2. What is an Analytical (OLAP) workload?

🔍 **Explanation:**OLAP systems are built for **reading large datasets**, doing aggregations, and uncovering trends — typically **not for real-time updates**.

Use Case:Netflix analyzes watch history across millions of users to personalize content. This doesn't need millisecond inserts, but rather powerful scanning and aggregations.

🛠️ **Example (OLAP-style query):**

```
-- Analyze monthly total spending per user
SELECT user_id, MONTH(timestamp) AS txn_month, SUM(amount) AS total_spend
FROM transactions
GROUP BY user_id, MONTH(timestamp);
```

3. What is a Primary Key, and why is it a real deal in Hybrid tables?

🔍 **Explanation:**A **Primary Key** is a column (or set of columns) that uniquely identifies every row in a table — no duplicates, no NULLs allowed.

🧾 **FinTech Example: Stripe** must ensure that every transaction has a unique `transaction_id` — it cannot afford to lose or overwrite records.

🛠️ **SQL Example:**

```
CREATE OR REPLACE HYBRID TABLE transactions (
  transaction_id STRING PRIMARY KEY,
  user_id STRING,
  amount NUMBER,
  timestamp TIMESTAMP
);
```

4. What is a Composite Primary Key and when is it useful?

🔍 **Explanation:**A **Composite Key** combines two or more columns to uniquely identify a row. Useful when no single field is unique on its own.

🏦 **Use Case:**In **Plaid**, a `user_id` and `bank_id` might form a composite key to uniquely identify a bank account across users and institutions.

🛠️ **SQL Example:**

```
CREATE OR REPLACE HYBRID TABLE user_bank_accounts (
  user_id STRING,
  bank_id STRING,
  account_type STRING,
  PRIMARY KEY (user_id, bank_id)
);
```

Composite keys prevent duplicate account entries for the same user and bank.

5. What is a Foreign Key and how does it enforce relationships?

🔍 **Explanation:** A **Foreign Key** links one table to another via a **primary key**, ensuring data integrity. It enforces that referenced data **must exist**.

🧾 **FinTech Example:** In **Robinhood**, each trade must refer to an existing user — a foreign key enforces this.

🛠️ **SQL Example:**

```
CREATE OR REPLACE HYBRID TABLE users (
  user_id STRING PRIMARY KEY,
  name STRING
);

CREATE OR REPLACE HYBRID TABLE trades (
  trade_id STRING PRIMARY KEY,
  user_id STRING,
  stock_symbol STRING,
  trade_amount NUMBER,
  FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

6. What is Referential Integrity and why is it critical?

🔍 **Explanation:Referential Integrity** ensures that the connection between tables is valid — no orphan records. You can't add a trade if the user doesn't exist.

🚫 **Invalid Insert:**

```
-- This will FAIL if 'usr_999' does not exist in users table
INSERT INTO trades VALUES ('trade_001', 'usr_999', 'AAPL', 1000);
```

This protects your data quality — crucial for auditability in finance.

7. What is an Index and how does it speed up queries?

🔍 **Explanation:**An **index** acts like a search engine. Without one, the system scans the entire table. With one, it goes straight to the data.

TECH-USE CASE:Amazon needs to retrieve orders by `user_id` instantly. Indexes make this fast.

🛠️ **SQL Example:**

```
-- Create index on user_id for quick filtering
CREATE INDEX idx_user_id ON transactions(user_id);
```

8. What's the problem with maintaining separate OLTP and OLAP systems?

🔍 **Explanation:**You need to build pipelines to move data between them — it causes:

- **Latency** in insights
- **Duplicated infrastructure**
- **Security headaches**

💼 **Real-world Example:Goldman Sachs** might move data from MongoDB (for transactions) to Snowflake (for analytics), delaying fraud detection. Unistore removes this barrier.

9. Why does real-time access matter in FinTech?

🔍 **Explanation:**Decisions like fraud detection or credit scoring must happen **now**, not later.

💳 **Example:**If **Zelle** delays detection, fraudulent transactions may be processed before detection kicks in. Unistore enables **immediate analytical insights** on real-time transactional data.

10. How does Snowflake unify OLTP and OLAP with Unistore?

🔍 **Explanation:**Unistore uses **Hybrid Tables** backed by a **row-based engine** for fast OLTP-like operations, but still **seamlessly integrates with columnar storage** for OLAP analytics.

🏗️ **Architecture Insight:**

- **Hybrid Tables:** Row-based, fast inserts/selects
- **Analytical Store:** Columnar, scalable queries
- **Shared Engine:** Same SQL, auth, metadata, compute
- **Invisible Replication:** No manual sync needed

Part 2: Deeper DISCUSSIONS About Unistore & Hybrid Tables (ETL, Architecture, Use)

1. How do Hybrid Tables work internally in Snowflake?

🔍 **Explanation:**Hybrid Tables:

- Use **row-based storage** → Fast for single inserts/updates
- Are **replicated automatically** to the analytic store
- Leverage **same compute/query engine** as Snowflake

🧠 **Difference:**Unlike traditional Snowflake tables, **Hybrid Tables don't use micro-partitions** (columnar), but **row storage** for OLTP efficiency.

🎯 **Benefit:**You don't need to choose "which engine" — it's automatically managed.

2. How can I build an end-to-end ETL/ELT pipeline with Unistore?

🔧 **Steps:**

- Load Raw Data into Hybrid Tables (e.g., payments, accounts)
- Transform with SQL (joins, filters, aggregations)
- Store or serve insights in regular Snowflake tables or dashboards
- Feed back transformed data to front-end apps in milliseconds

🛠️ **SQL Example:**

```
-- Load transactions
INSERT INTO hybrid_transactions VALUES (...);

-- Transform: Aggregate user daily spend
CREATE OR REPLACE TABLE daily_summary AS
SELECT user_id, DATE(timestamp) AS txn_date, SUM(amount) AS daily_spend
FROM hybrid_transactions
GROUP BY user_id, DATE(timestamp);
```

3. What use cases does Unistore unlock that weren't possible before?

- Real-time fraud detection using current + historical data
- In-app analytics dashboards powered directly from transactional tables
- Alerting systems reacting instantly to operational data

🏦 **Example:SoFi** could detect a credit threshold breach and instantly trigger UI alerts + credit score checks — all using Unistore without syncing across DBs.

4. Can Hybrid Tables be used in joins with regular Snowflake tables?

✅ **Yes!**Snowflake allows **joins between Hybrid and regular tables**, treating them as a **single logical dataset**.

🛠️ **SQL Example:**

```
SELECT t.user_id, u.name, t.amount
FROM hybrid_transactions t
JOIN users u ON t.user_id = u.user_id;
```

5. How does Unistore simplify app development in FinTech or TECH environments?

Explanation: Developers can now:

- Store real-time app data in Hybrid Tables
- Analyze instantly in dashboards
- Trigger alerts or update workflows
- Eliminate syncing layers like Kafka → ETL → Snowflake

🔁 **Before Unistore:**Frontend App → OLTP DB → ETL → Snowflake → BI Tools

🌐 **With Unistore:**Frontend App ↔ **Snowflake only** (via Hybrid Tables)

🧠 **Example:Uber's** driver payment app could run entirely on Snowflake, storing trips as Hybrid Table records and instantly calculating payouts.

💼 Use Case: Real-time Credit Breach Detection & Alert System in PayPal using Snowflake Unistore

🧠 **Business Scenario** PayPal wants to:

- **Track customer credit usage in real time**
- Instantly **alert users** when they exceed their allowed credit
- Provide **dashboards to analysts** showing breach trends and usage patterns

This must be built on **one system** without using:

- An external OLTP DB (like PostgreSQL)
- Middleware queues (Kafka)
- Separate ETL tools

Unistore + Hybrid Tables in Snowflake makes this possible.

🏗️ System Architecture Overview

🧩 Components:

1. `customers` → Static metadata table (regular Snowflake table)
2. `hybrid_transactions` → Live credit transactions (Hybrid Table)
3. `credit_breach_alerts` → Alert log (Hybrid Table)
4. `daily_summary` → Aggregated analytics (Regular Table or Materialized View)
5. **SQL logic / tasks / streams** → For alerting and transformation

Step 1: Set up base tables

A. Customer Master Table (Regular Table)

```
CREATE OR REPLACE TABLE customers (
  customer_id STRING PRIMARY KEY,
  name STRING,
  credit_limit NUMBER
);

-- Sample data
INSERT INTO customers VALUES
  ('CUST001', 'Alice', 1000),
  ('CUST002', 'Bob', 500),
  ('CUST003', 'Charlie', 750);
```

B. Transactions Table (HYBRID TABLE)

```
CREATE OR REPLACE HYBRID TABLE hybrid_transactions (
  txn_id STRING PRIMARY KEY,
  customer_id STRING,
  amount NUMBER,
  txn_time TIMESTAMP,
  FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);

-- Simulate incoming transactions
INSERT INTO hybrid_transactions VALUES
  ('TXN1001', 'CUST001', 300, CURRENT_TIMESTAMP),
  ('TXN1002', 'CUST001', 400, CURRENT_TIMESTAMP),
  ('TXN1003', 'CUST002', 600, CURRENT_TIMESTAMP); -- breach
```

✅ **Referential Integrity is enforced: no transaction for non-existing customers**

Step 2: Alert Table for Credit Breach (HYBRID TABLE)

```
CREATE OR REPLACE HYBRID TABLE credit_breach_alerts (
  alert_id STRING PRIMARY KEY,
  customer_id STRING,
  total_spent NUMBER,
  credit_limit NUMBER,
  breach_time TIMESTAMP
);
```

Step 3: Create logic to detect breach (ELT with INSERT-SELECT)

💡 **This logic compares a customer's total spend in the hybrid_transactions table against their credit limit in customers**.

```
-- Insert into alerts table if spending > credit limit
INSERT INTO credit_breach_alerts
SELECT
  UUID_STRING() AS alert_id,
  t.customer_id,
  SUM(t.amount) AS total_spent,
  c.credit_limit,
  CURRENT_TIMESTAMP
FROM hybrid_transactions t
JOIN customers c ON t.customer_id = c.customer_id
GROUP BY t.customer_id, c.credit_limit
HAVING SUM(t.amount) > c.credit_limit;
```

📈 **Step 4: Analytical Summary for Dashboard:** This could go to a dashboard like Looker/Tableau/Streamlit.

```
-- Daily spend summary for internal insights
CREATE OR REPLACE TABLE daily_summary AS
SELECT
  customer_id,
  DATE(txn_time) AS txn_date,
  SUM(amount) AS daily_spend,
  COUNT(*) AS txn_count
FROM hybrid_transactions
GROUP BY customer_id, DATE(txn_time);
```

🔁 Optional: Automate using TASKS

To auto-run the alert logic every minute (real-time reaction):

```
CREATE OR REPLACE TASK detect_credit_breach
WAREHOUSE = compute_wh
SCHEDULE = '1 MINUTE'
AS
INSERT INTO credit_breach_alerts
SELECT
  UUID_STRING(), t.customer_id, SUM(t.amount), c.credit_limit,
CURRENT_TIMESTAMP
FROM hybrid_transactions t
JOIN customers c ON t.customer_id = c.customer_id
GROUP BY t.customer_id, c.credit_limit
HAVING SUM(t.amount) > c.credit_limit;
```

✅ Outcome of the Pipeline

Component	Type	Function
`customers`	Regular table	Holds metadata, like credit limit
`hybrid_transactions`	Hybrid Table	Stores real-time transactional data
`credit_breach_alerts`	Hybrid Table	Triggers & logs alerts immediately
`daily_summary`	Analytical table	Power dashboard metrics
`detect_credit_breach`	Task	Automates detection every minute

🎯 Real-World Benefits for PayPal

Feature	Benefit
💡 Hybrid Tables	Fast inserts for real-time processing
🔒 Referential Integrity	Guarantees data quality (no orphan txns)
📊 Unified Store	No separate ETL or DB needed
🔁 Automation	Auto-monitor credit and send alerts instantly
⚡ Instant Analytics	Finance & Risk teams see live trends

🧪 What you learnt in this Lab: If you're onboarding a **FinTech Data Engineer**, this lab:

- Teaches OLTP + OLAP unification
- Introduces real-time alerting logic
- Uses just Snowflake SQL (no third-party tools)
- Is extendable: Add email alerts via Snowflake External Functions or Streamlit UIs for visual alerts

Happy learning
Best regards
Saransh Jain