

# Streams and Tasks In Snowflake

## 1. Streams :-

- Streams are used to capture the DML operations like insert, update, delete.
- If we define a stream object for a table then it will capture whatever changes happened on that particular table like update, insert and delete.
- Meta data action column in the stream table will store the actions like insert, delete etc.

I have created a table called `player_table` as a stage table and `player` as a final or destination table

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' tab is active, showing a tree view of the database structure. The 'TRAINING' database is selected, and the 'PUBLIC' schema is expanded, showing a list of tables including 'ITEM\_1', 'ITEM\_STAGE\_1', 'PLAYER', and 'PLAYER\_TABLE'. The main editor area displays the following SQL code:

```
1 //staging table
2 CREATE OR REPLACE TABLE PLAYER_TABLE(
3   ID INT,
4   NAME VARCHAR(20),
5   SCORE INT
6 );
7
8 //destination table
9 CREATE OR REPLACE TABLE PLAYER (
10  ID INT,
11  NAME VARCHAR(20),
12  SCORE INT,
13  INSERT_DATE_TIME TIMESTAMP_NTZ(9) DEFAULT CURRENT_TIMESTAMP (),
14  UPDATE_DATE_TIME TIMESTAMP_NTZ(9)
15 );
```

The 'Results' tab is active, showing a single row with the status 'Table PLAYER successfully created.' The 'Query Details' panel on the right shows the query duration as 156ms, 1 row, and the query ID as 01b07dde-3200-f05d-0...

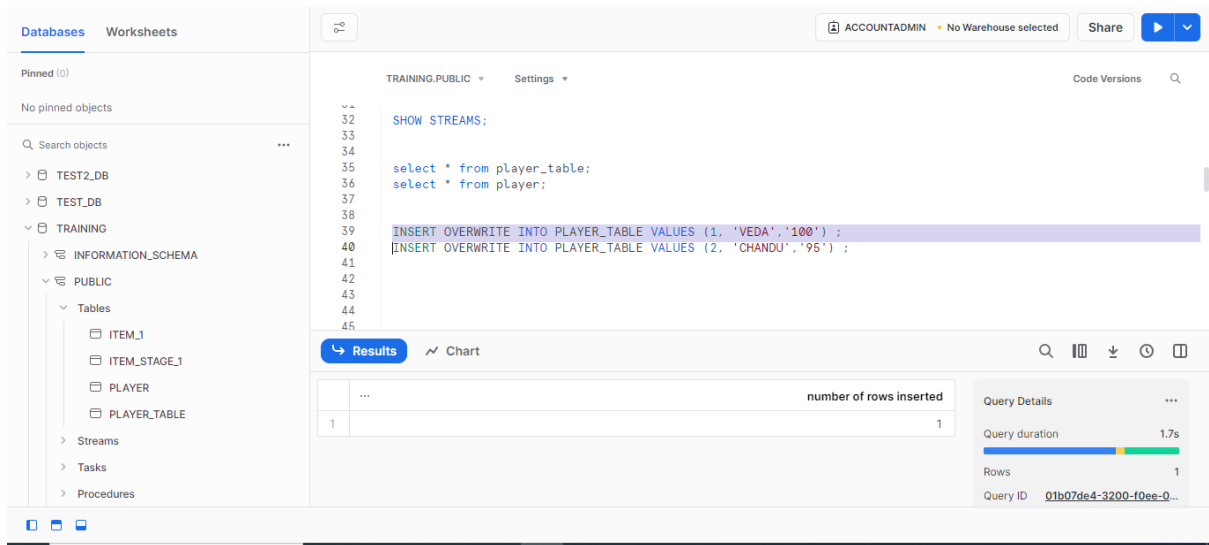
Now I have created a stream on stage table called `player_table` in order to capture the changes occurred on stage table

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' tab is active, showing a tree view of the database structure. The 'TRAINING' database is selected, and the 'PUBLIC' schema is expanded, showing a list of tables including 'ITEM\_1', 'ITEM\_STAGE\_1', 'PLAYER', and 'PLAYER\_TABLE'. The main editor area displays the following SQL code:

```
20
21 select * from player_table;
22 select * from player;
23
24 CREATE OR REPLACE STREAM PLAYER_UPLOAD_STREAM
25 ON TABLE PLAYER_TABLE;
```

The 'Results' tab is active, showing a single row with the status 'Stream PLAYER\_UPLOAD\_STREAM successfully created.' The 'Query Details' panel on the right shows the query duration as 271ms, 1 row, and the query ID as 01b07dde-3200-f05d-0...

There is no data available in the both tables now I am adding one record in the stage table



The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' tab is active, showing a tree view of the database structure. The 'TRAINING' database is expanded, showing the 'PUBLIC' schema and its tables: 'ITEM\_1', 'ITEM\_STAGE\_1', 'PLAYER', and 'PLAYER\_TABLE'. The main editor area shows a SQL query with line numbers 32 to 45. The query is:

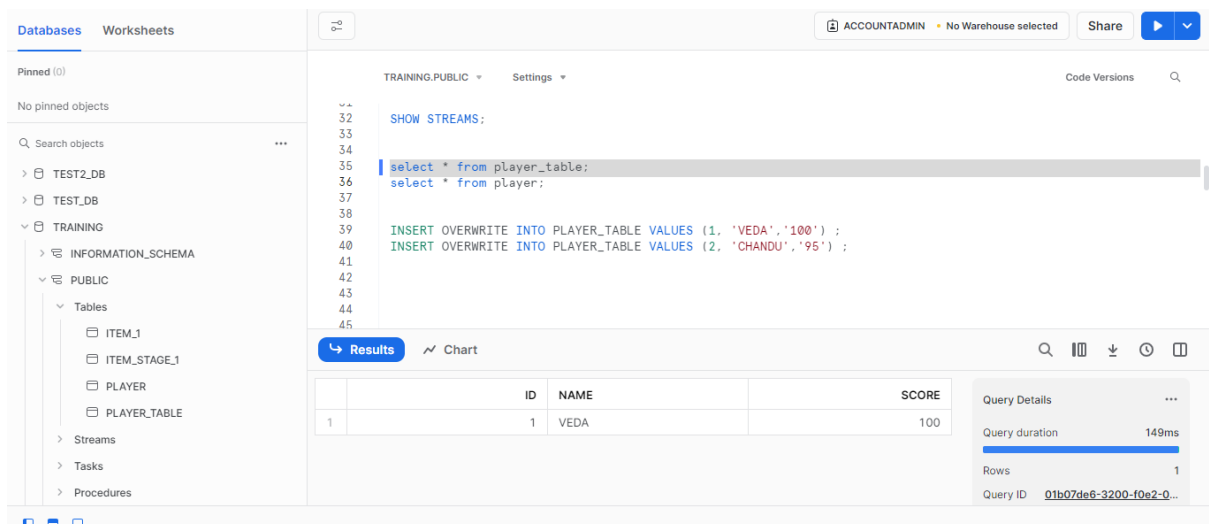
```
SHOW STREAMS;

select * from player_table;
select * from player;

INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA','100') ;
INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU','95') ;
```

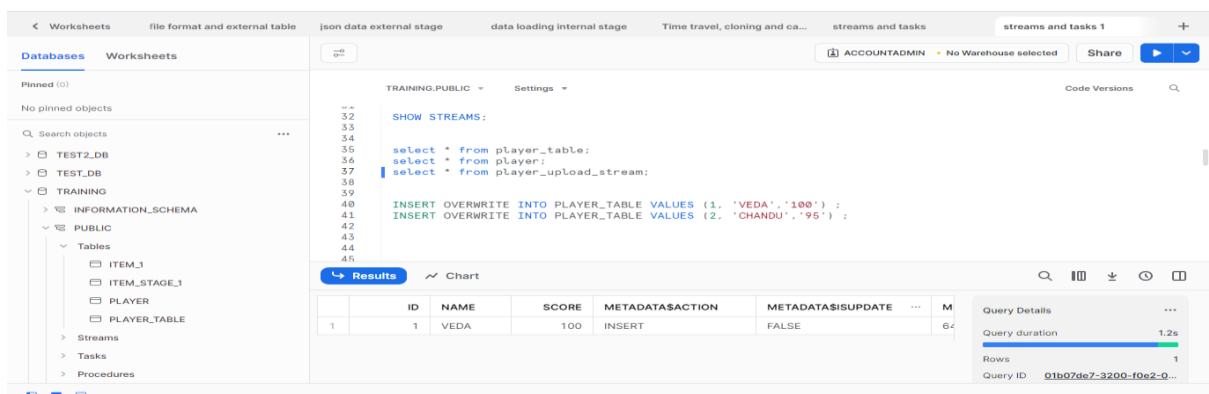
The 'Results' tab is selected, showing a table with one row and one column, 'number of rows inserted', with the value '1'. The 'Query Details' panel on the right shows 'Query duration' as 1.7s, 'Rows' as 1, and 'Query ID' as 01b07de4-3200-f0ee-0...

Now there is one record in player\_table



The screenshot shows the Snowflake SQL Editor interface. The 'Databases' tab is active, showing the same tree view as before. The main editor area shows the same SQL query as before. The 'Results' tab is selected, showing a table with three columns: 'ID', 'NAME', and 'SCORE'. The table contains one row with the values (1, VEDA, 100). The 'Query Details' panel on the right shows 'Query duration' as 149ms, 'Rows' as 1, and 'Query ID' as 01b07de6-3200-f0e2-0...

Now this action needs to be captured in stream table, so insert operation has been recorded.



The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' tab is active, showing the same tree view as before. The main editor area shows a SQL query with line numbers 32 to 45. The query is:

```
SHOW STREAMS;

select * from player_table;
select * from player;
select * from player_upload_stream;

INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA','100') ;
INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU','95') ;
```

The 'Results' tab is selected, showing a table with six columns: 'ID', 'NAME', 'SCORE', 'METADATA\$ACTION', 'METADATA\$ISUPDATE', and 'METADATA\$ROWID'. The table contains one row with the values (1, VEDA, 100, INSERT, FALSE, 64). The 'Query Details' panel on the right shows 'Query duration' as 1.2s, 'Rows' as 1, and 'Query ID' as 01b07de7-3200-f0e2-0...

Now I am overwriting this one record with another record

TRAINING.PUBLIC ▾ Settings ▾

```
38  
39  
40  
41 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA', '100') ;  
42 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU', '95') ;  
43  
44  
45  
46  
47  
48  
49  
50  
51 INSERT OVERWRITE INTO ITEM_STAGE_1 VALUES (3, 'Product3', 'Category40') ;
```

Results Chart

	number of rows inserted
1	1

Query Details

- Query duration: 1.5s
- Rows: 1
- Query ID: 01b07de8-3200-f05d-0...

So, the values in a record will be changed now

TRAINING.PUBLIC ▾ Settings ▾

```
33  
34  
35  
36 select * from player_table;  
37 select * from player;  
38 select * from player_upload_stream;  
39  
40  
41 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA', '100') ;  
42 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU', '95') ;  
43  
44  
45  
46
```

Results Chart

	ID	NAME	SCORE
1	2	CHANDU	95

Query Details

- Query duration: 97ms
- Rows: 1
- Query ID: 01b07de9-3200-f0e2-0...

If we overwrite the data in stage table then the record in the stream table also overwrites.

TRAINING.PUBLIC ▾ Settings ▾

```
33  
34  
35  
36 select * from player_table;  
37 select * from player;  
38 select * from player_upload_stream;  
39  
40  
41 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA', '100') ;  
42 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU', '95') ;  
43  
44  
45  
46
```

Results Chart

	ID	NAME	SCORE	METADATA\$ACTION	METADATA\$ISUPDATE	M
1	2	CHANDU	95	INSERT	FALSE	14

Query Details

- Query duration: 1.4s
- Rows: 1
- Query ID: 01b07de9-3200-f0da-0...

I have created a stored procedure like a function in order to merge the captured data into the final table using stream and have written conditional statements.

### QUERY :-

```
CREATE OR REPLACE PROCEDURE "ITEM_MERGE"()
RETURNS VARCHAR (16777216)
LANGUAGE SQL
EXECUTE AS OWNER
AS
$$
BEGIN

    MERGE INTO TRAINING.PUBLIC. PLAYER p USING TRAINING.PUBLIC.
    PLAYER_UPLOAD_STREAM ps
    ON p.ID = ps.ID
    WHEN MATCHED AND METADATA$action = 'INSERT'
    THEN UPDATE SET p.NAME = ps.NAME,
        p.SCORE = ps.SCORE,
        p.UPDATE_DATE_TIME = CURRENT_TIMESTAMP()
    WHEN NOT MATCHED AND METADATA$action = 'INSERT'
    THEN INSERT (p.ID, p.NAME, p.SCORE )
        VALUES (ps.ID, ps.NAME, ps.SCORE);

Return 'Success';

END;

$$;

call ITEM_MERGE ();
```

**Databases Worksheets** ACCOUNTADMIN COMPUTE\_WH Share

TRAINING.PUBLIC Settings Code Versions

```

48 // Stored Procedure
49 CREATE OR REPLACE PROCEDURE "ITEM_MERGE"()
50 RETURNS VARCHAR(16777216)
51 LANGUAGE SQL
52 EXECUTE AS OWNER
53 AS
54 $$
55 BEGIN
56     MERGE INTO TRAINING.PUBLIC.PLAYER p USING TRAINING.PUBLIC.PLAYER_UPLOAD_STREAM ps
57     ON p.ID = ps.ID
58     WHEN MATCHED AND METADATA$ACTION = 'INSERT'
59     THEN UPDATE SET p.NAME = ps.NAME,
60                  p.SCOR = ps.SCOR,
61                  p.UPDATE_DATE_TIME = CURRENT_TIMESTAMP()

```

**Results** Chart

status
Function ITEM_MERGE successfully created.

**Query Details**

Query duration: 54ms

Rows: 1

Query ID: 01b07df2-3200-f0ee-0...

Now I am calling a function and now the values in the stream will be merged into destination table.

**Databases Worksheets** ACCOUNTADMIN COMPUTE\_WH Share

TRAINING.PUBLIC Settings Code Versions

```

64     VALUES (ps.ID, ps.NAME, ps.SCOR);
65
66     Return 'Success';
67
68 END;
69 $$
70
71
72 call ITEM_MERGE();
73
74
75
76
77 -- 1 minute is 60 seconds, so 5 seconds is 5/60 hours

```

**Results** Chart

ITEM_MERGE
Success

**Query Details**

Query duration: 619ms

Rows: 1

Query ID: 01b07df2-3200-f0da-0...

Now the record has been merged into player and insert time stamp will be recorded and update timestamp here will be null.

**Databases Worksheets** ACCOUNTADMIN COMPUTE\_WH Share

TRAINING.PUBLIC Settings Code Versions

```

31 SHOW STREAMS;
32
33
34
35 select * from player_table;
36 select * from player;
37 select * from player_upload_stream;
38
39
40 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA', '100') ;
41 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU', '95') ;
42 INSERT OVERWRITE INTO ITEM_STAGE_1 VALUES (3, 'Product3', 'Category40') ;
43
44

```

**Results** Chart

	ID	NAME	SCORE	...	INSERT_DATE_TIME	UPDATE_DATE_TIME
1	2	CHANDU	95		2023-11-21 20:34:52.970	null

**Query Details**

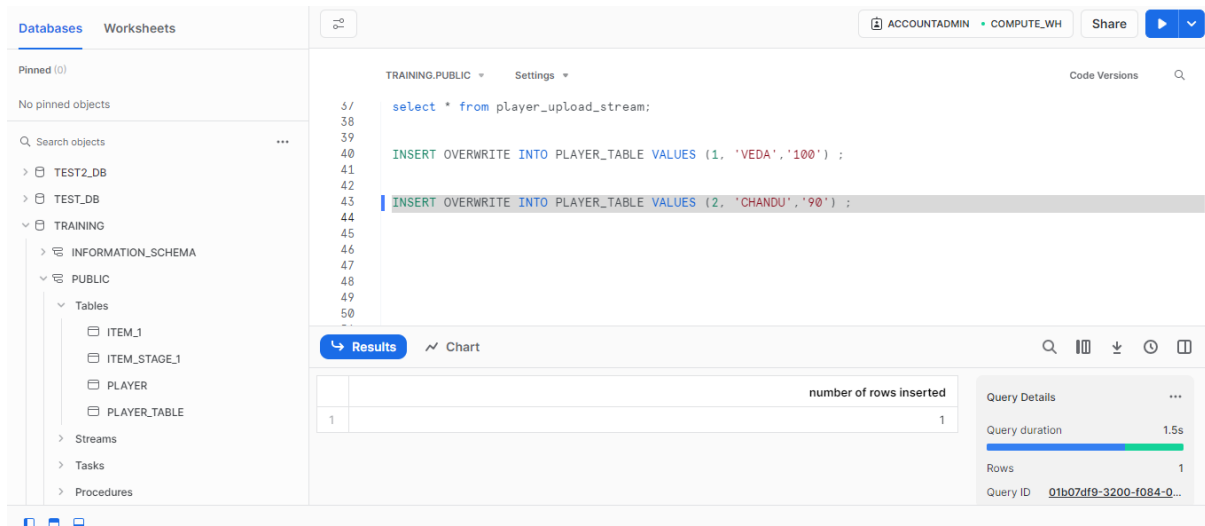
Query duration: 134ms

Rows: 1

Query ID: 01b07df7-3200-f0ee-0...

If I update the record then the update time stamp will be recorded too

I am updating the score from 95 to 90



The screenshot shows a database interface with a SQL editor and a results pane. The SQL editor contains the following query:

```
select * from player_upload_stream;

INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA','100') ;

INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU','90') ;
```

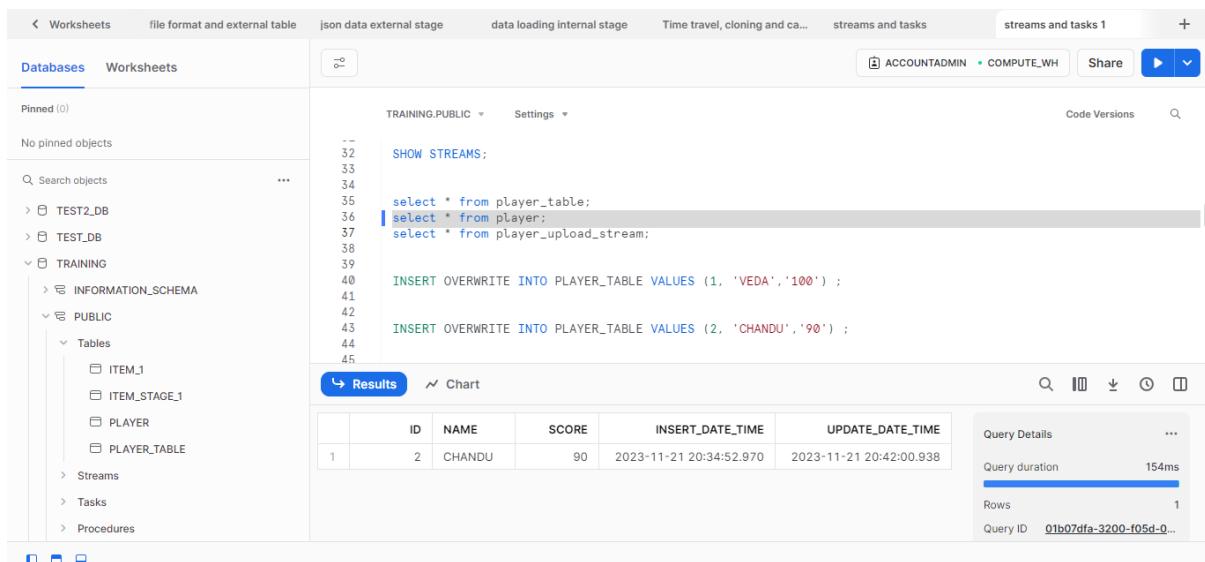
The results pane shows a table with one row:

number of rows inserted
1

Query Details:

- Query duration: 1.5s
- Rows: 1
- Query ID: 01b07df9-3200-f084-0...

Now I am calling the item\_merge function and then we can see the changes occurred in destination table and we can see the update timestamp here



The screenshot shows a database interface with a SQL editor and a results pane. The SQL editor contains the following query:

```
--
SHOW STREAMS;

select * from player_table;
select * from player;
select * from player_upload_stream;

INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA','100') ;

INSERT OVERWRITE INTO PLAYER_TABLE VALUES (2, 'CHANDU','90') ;
```

The results pane shows a table with one row:

	ID	NAME	SCORE	INSERT_DATE_TIME	UPDATE_DATE_TIME
1	2	CHANDU	90	2023-11-21 20:34:52.970	2023-11-21 20:42:00.938

Query Details:

- Query duration: 154ms
- Rows: 1
- Query ID: 01b07dfa-3200-f05d-0...

Now I am creating a task to run the particular function at regular intervals

I am creating a task for stream table in order to merge whenever the data is present in stage table it needs to move or merge that record information to the final table

I have scheduled the interval for 1 minute

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' tab is active, showing a tree view of the database structure: TEST2\_DB, TEST\_DB, and TRAINING. Under TRAINING, there is an INFORMATION\_SCHEMA and a PUBLIC schema containing tables ITEM\_1, ITEM\_STAGE\_1, PLAYER, and PLAYER\_TABLE. The main editor area shows the following SQL code:

```
8/
88
//user-managed task
89 CREATE OR REPLACE TASK PUSH_PLAYER_UPLOADS_TASK
90 WAREHOUSE = 'COMPUTE_WH'
91 SCHEDULE = '1 minute'
92 WHEN
93 SYSTEM$STREAM_HAS_DATA('PLAYER_UPLOAD_STREAM')
94 AS CALL ITEM_MERGE();
95
96
97 ALTER TASK PUSH_ITEM_UPLOADS_TASK SUSPEND;
98
99
100 ALTER TASK PUSH_ITEM_UPLOADS_TASK RESUME;
```

Below the code editor, the 'Results' tab is active, displaying a single row with the status: 'Task PUSH\_PLAYER\_UPLOADS\_TASK successfully created.' The 'Query Details' panel on the right shows a query duration of 109ms, 1 row, and a query ID of 01b07e18-3200-f0da-0...

Now for every minute the task runs and it shows the status as scheduled.

And if the data is present in stage table and moved to final table then it will show the status or task history as succeeded after 1 minute if not it will show as skipped.

Now I have inserted or overwritten the one record

The screenshot shows the Snowflake SQL Editor interface with the 'Streams and tasks 1' worksheet selected. The main editor area shows the following SQL code:

```
33
34
35 select * from player_table;
36 select * from player;
37 select * from player_upload_stream;
38
39
40 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (1, 'VEDA', '100') ;
41
42
43 INSERT OVERWRITE INTO PLAYER_TABLE VALUES (3, 'VEDAMALIKA', '100') ;
44
45
46
```

Below the code editor, the 'Results' tab is active, displaying a single row with the number of rows inserted: 1. The 'Query Details' panel on the right shows a query duration of 370ms, 1 row, and a query ID of 01b07e1e-3200-f0e2-0...

Now I have resumed the task and then the task is scheduled

Query :-

**ALTER TASK PUSH\_player\_UPLOADS\_TASK RESUME;**

The screenshot shows a database interface with a sidebar on the left containing a tree view of databases and tables. The main area displays a SQL query editor with the following code:

```
yy
100 ALTER TASK PUSH_player_UPLOADS_TASK RESUME;
101
102 SHOW TASKS;
103
104
105
106
107
108
109
110
111
112
```

Below the query editor, the 'Results' tab is active, showing a table with the following data:

		STATE	ERROR_CODE	ERROR_MESSAGE	SCHEDULED
1	LOAD_STREAM)	SCHEDULED	null	null	2023-11-21 21:20:35.5

On the right side, the 'Query Details' panel shows the following information:

- Query duration: 279ms
- Rows: 1
- Query ID: 01b07e1f-3200-f0ee-0...

Now the schedule has been succeeded whenever I have added the data and skipped when there is no data

The screenshot shows the same database interface as the previous one, but with a different SQL query and results. The query editor contains:

```
yy
100 ALTER TASK PUSH_PLAYER_UPLOADS_TASK RESUME;
101
102 SHOW TASKS;
103
104
105
106
107
108
109
110
111
112
```

The 'Results' tab shows a table with the following data:

		STATE	ERROR_CODE	ERROR_MESSAGE	
1	LOAD_STREAM)	SCHEDULED	null	null	2
2	_STREAM)	SKIPPED	0040003	Conditional expression for task evaluated to false.	2
3	_STREAM)	SUCCEEDED	null	null	2
4	_STREAM)	SUCCEEDED	null	null	2

The 'Query Details' panel on the right shows:

- Query duration: 307ms
- Rows: 4
- Query ID: 01b07e22-3200-f0ee-0...