DDL commands

RUN | datarun | view | AV | R4 | 3NF | bcd | RUN1 | HR_U

Limit to 1000 rows

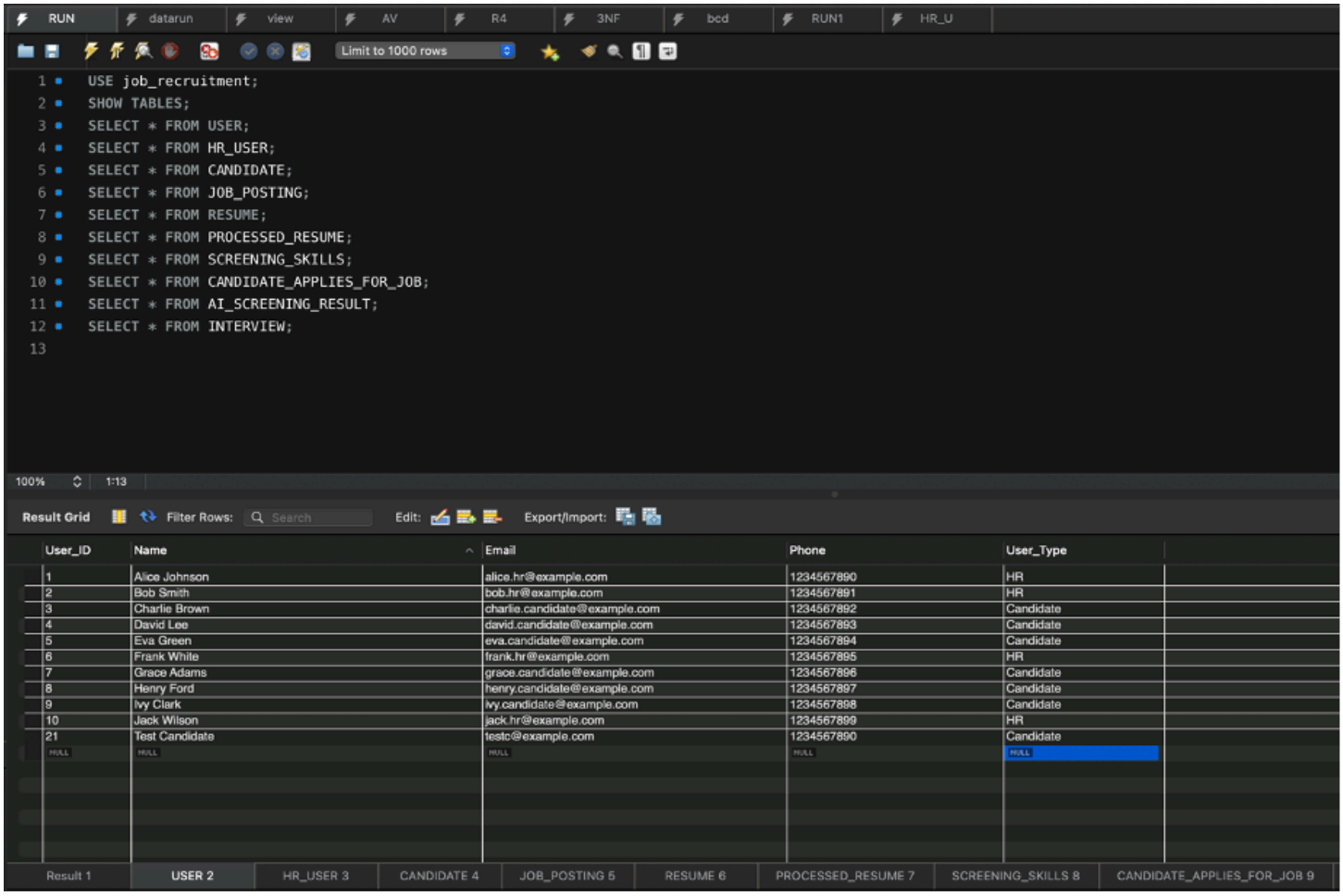
```
1  USE job_recruitment;
2  SHOW TABLES;
3  SELECT * FROM USER;
4  SELECT * FROM HR_USER;
5  SELECT * FROM CANDIDATE;
6  SELECT * FROM JOB_POSTING;
7  SELECT * FROM RESUME;
8  SELECT * FROM PROCESSED_RESUME;
9  SELECT * FROM SCREENING_SKILLS;
10 SELECT * FROM CANDIDATE_APPLIES_FOR_JOB;
11 SELECT * FROM AI_SCREENING_RESULT;
12 SELECT * FROM INTERVIEW;
13
```

100% 1:13

Result Grid | Filter Rows: Search | Export:

Tables_in_job_recruitment
AI_SCREENING_RESULT
AI_SCREENING_RESULT_3NF
AI_SCREENING_RESULT_RAW
CANDIDATE
CANDIDATE_APPLIES_FOR_JOB
CANDIDATE_INFO
HR_USER
INTERVIEW
JOB
JOB_POSTING
JOB_REQUIRED_SKILL
PROCESSED_RESUME
RESUME
SCREENING_SKILLS
UnnormalizedOrders
USER

Fig 1 Creating database and Creating Tables



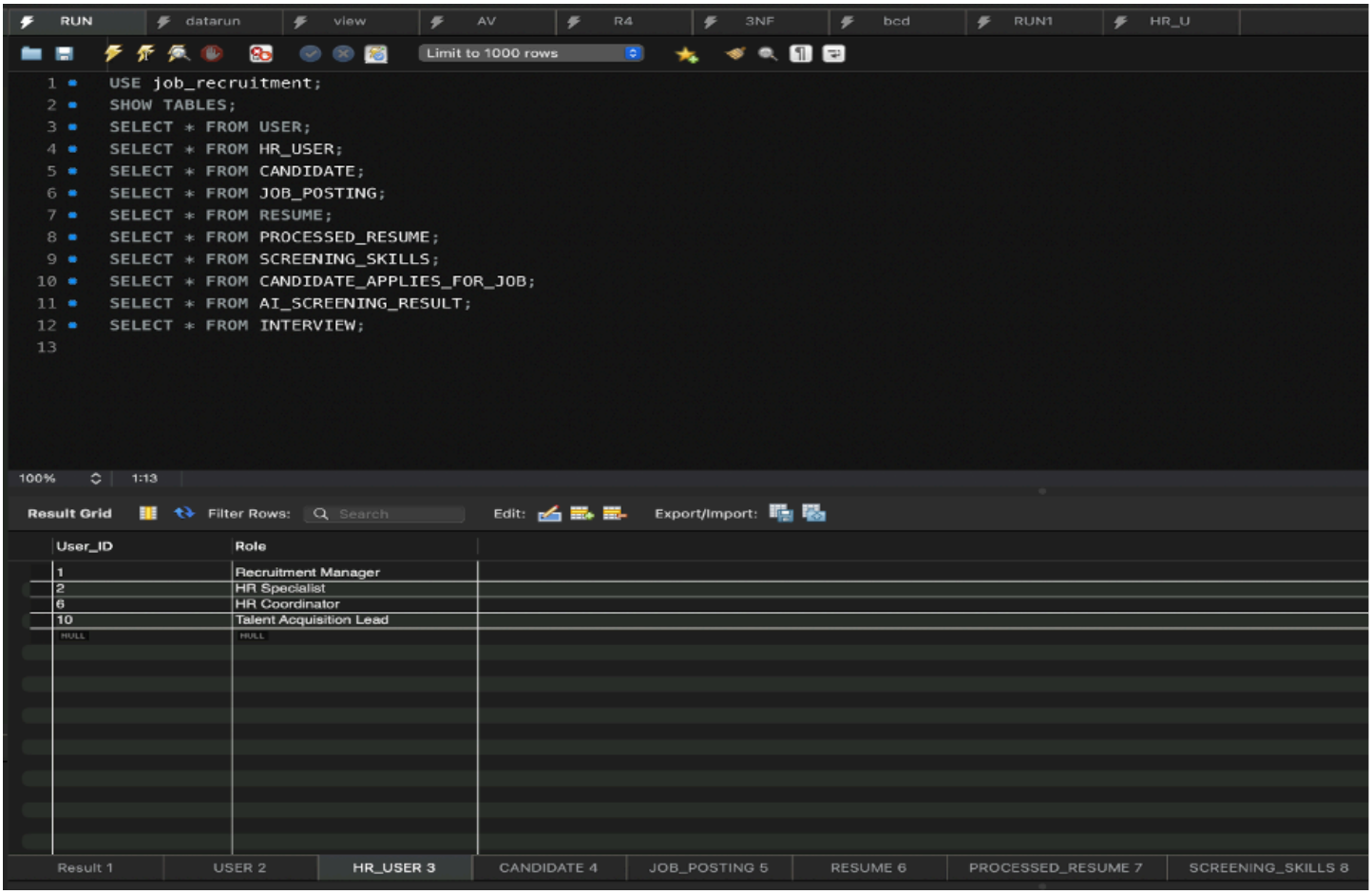
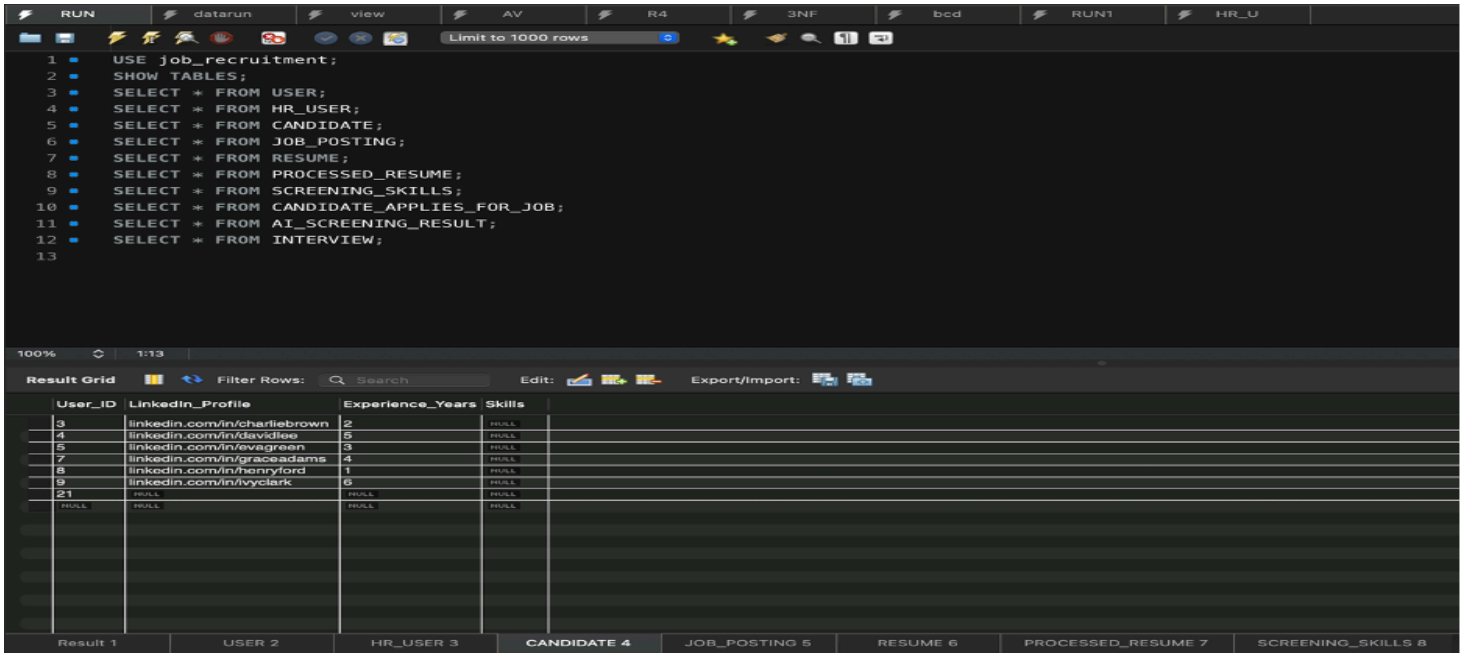


Fig 2 Creating USER and HR_USER and inserting values



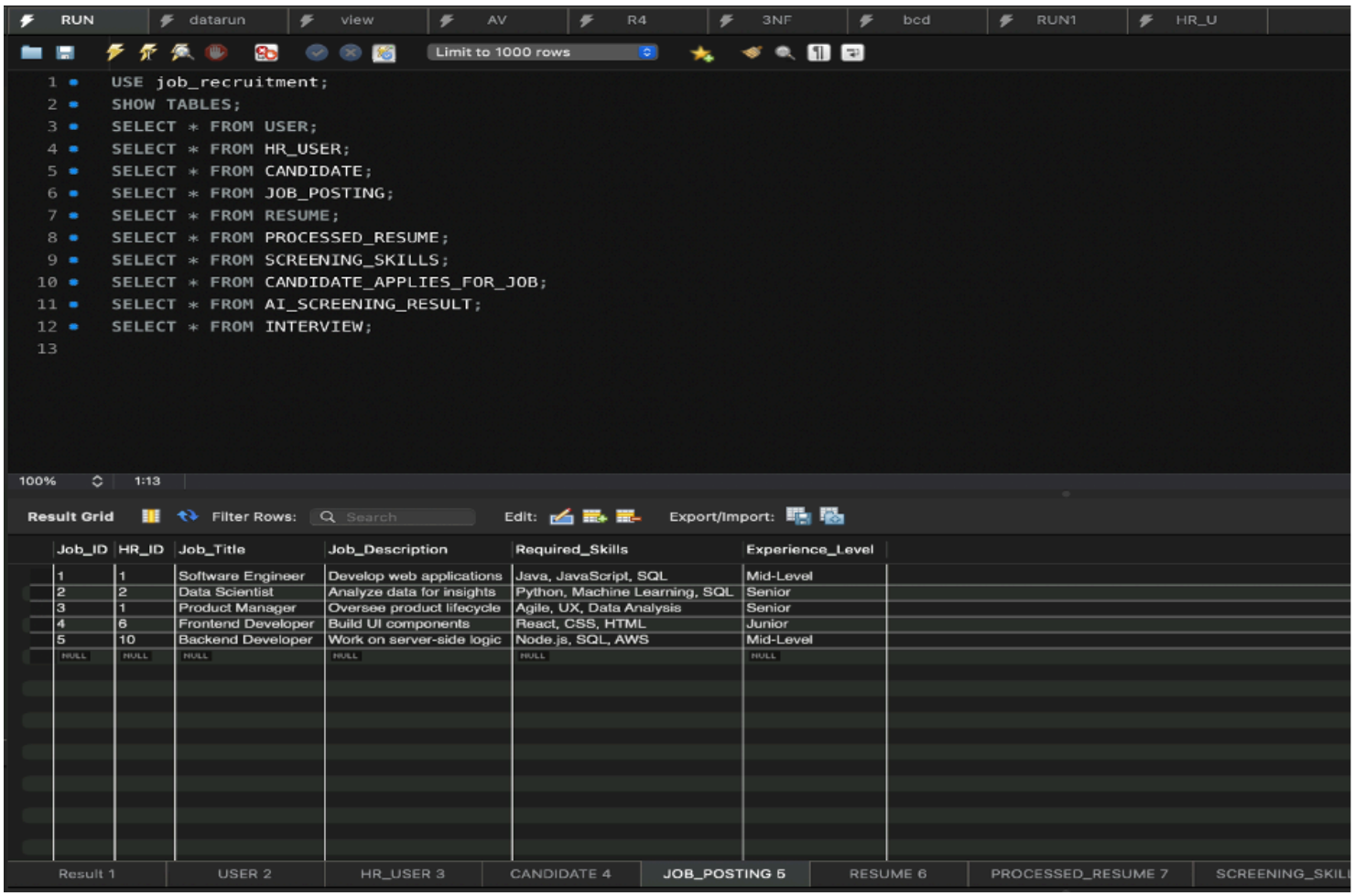
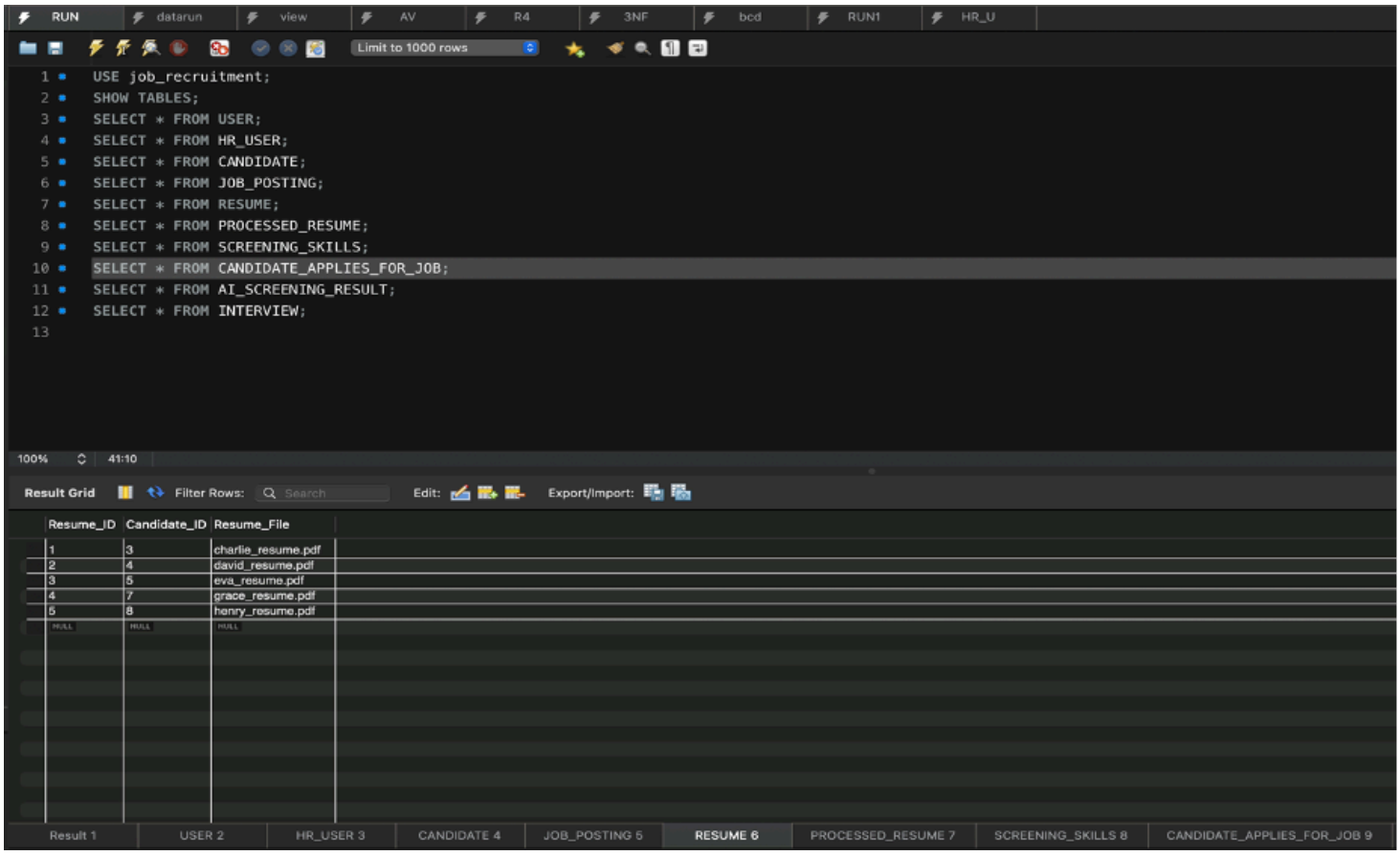


Fig 3 Creating CANDIDATE and JOB_POSTING and inserting values



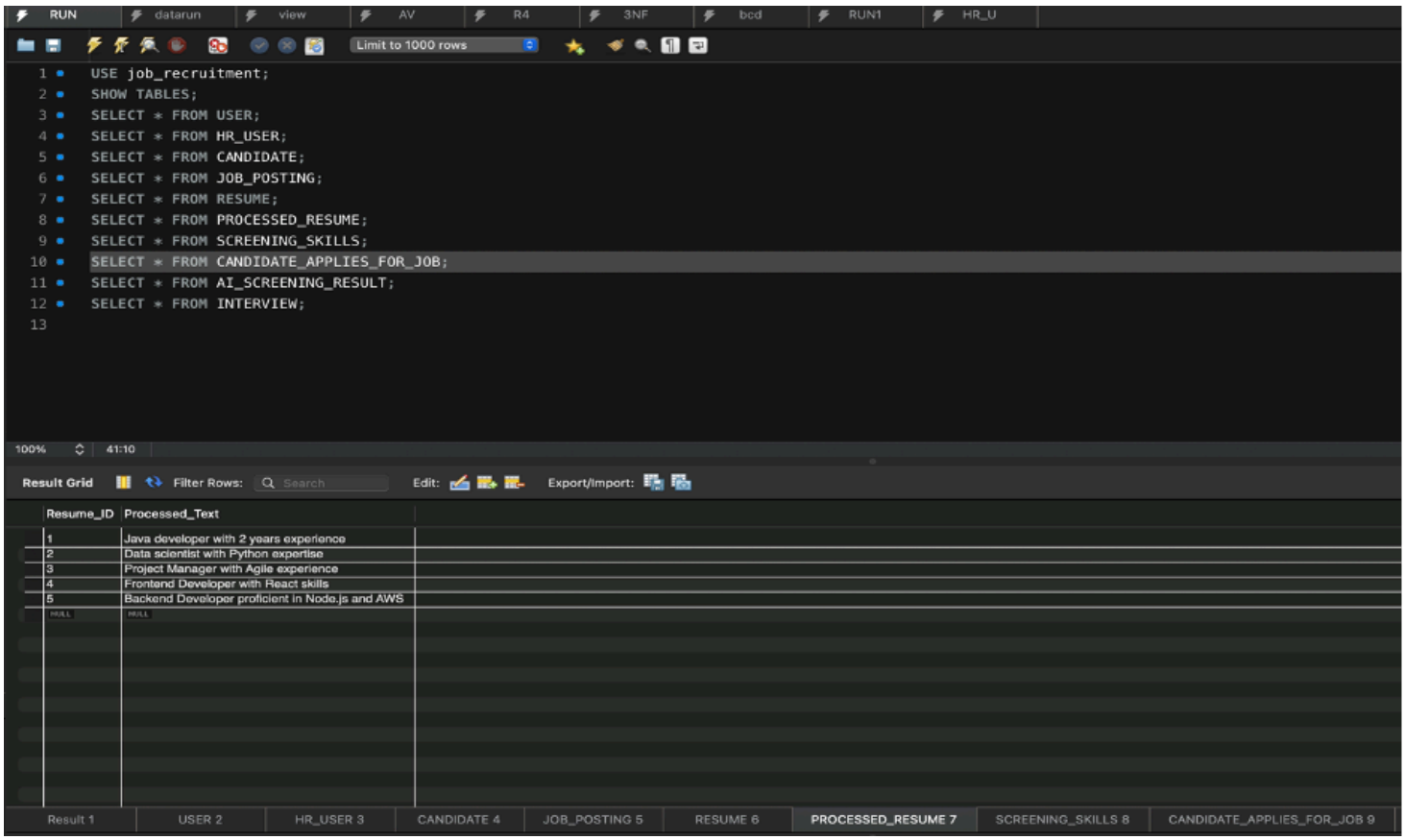
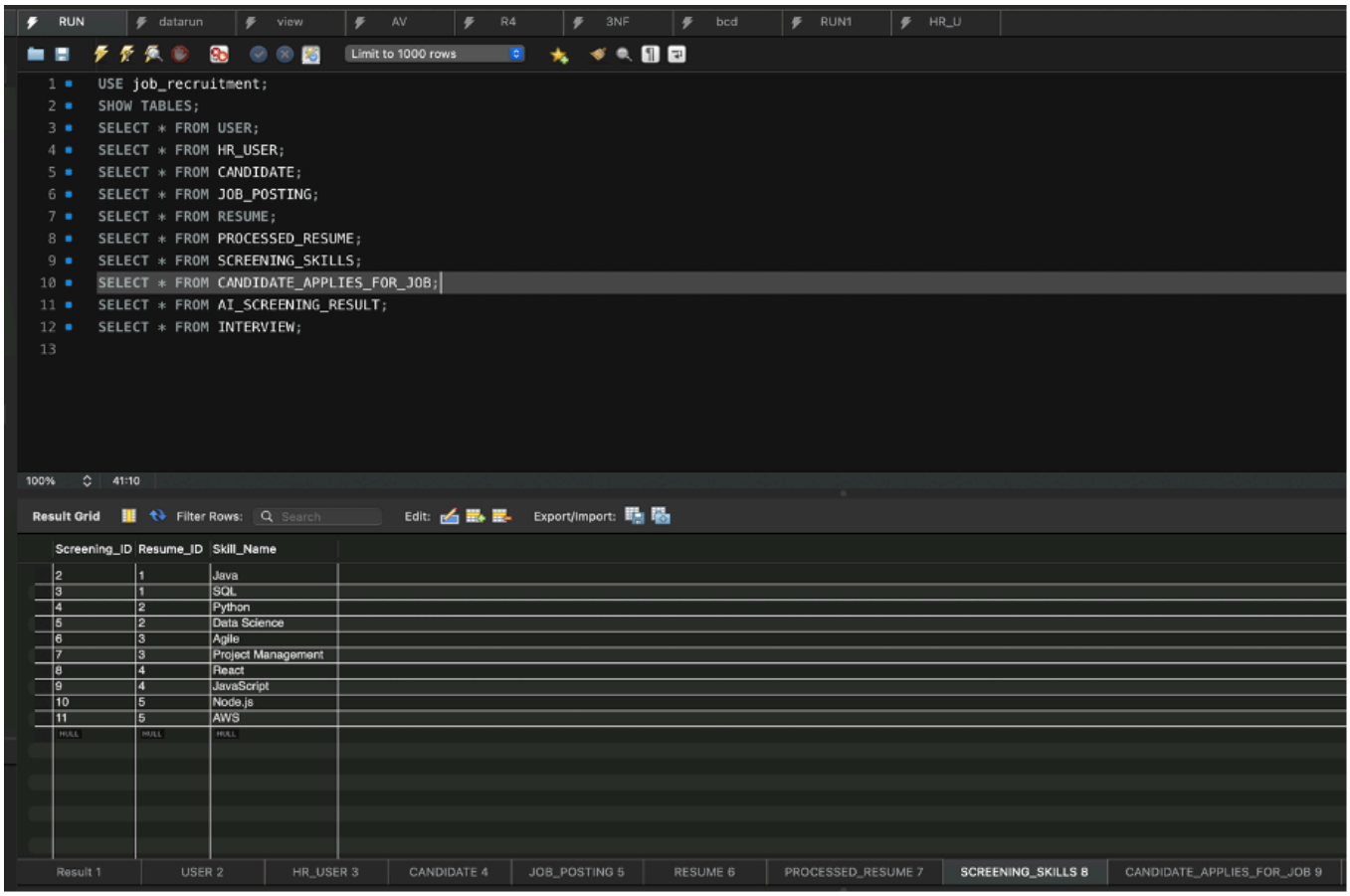


Fig 4 Creating table RESUME and PROCESSED_RESUME and inserting the values



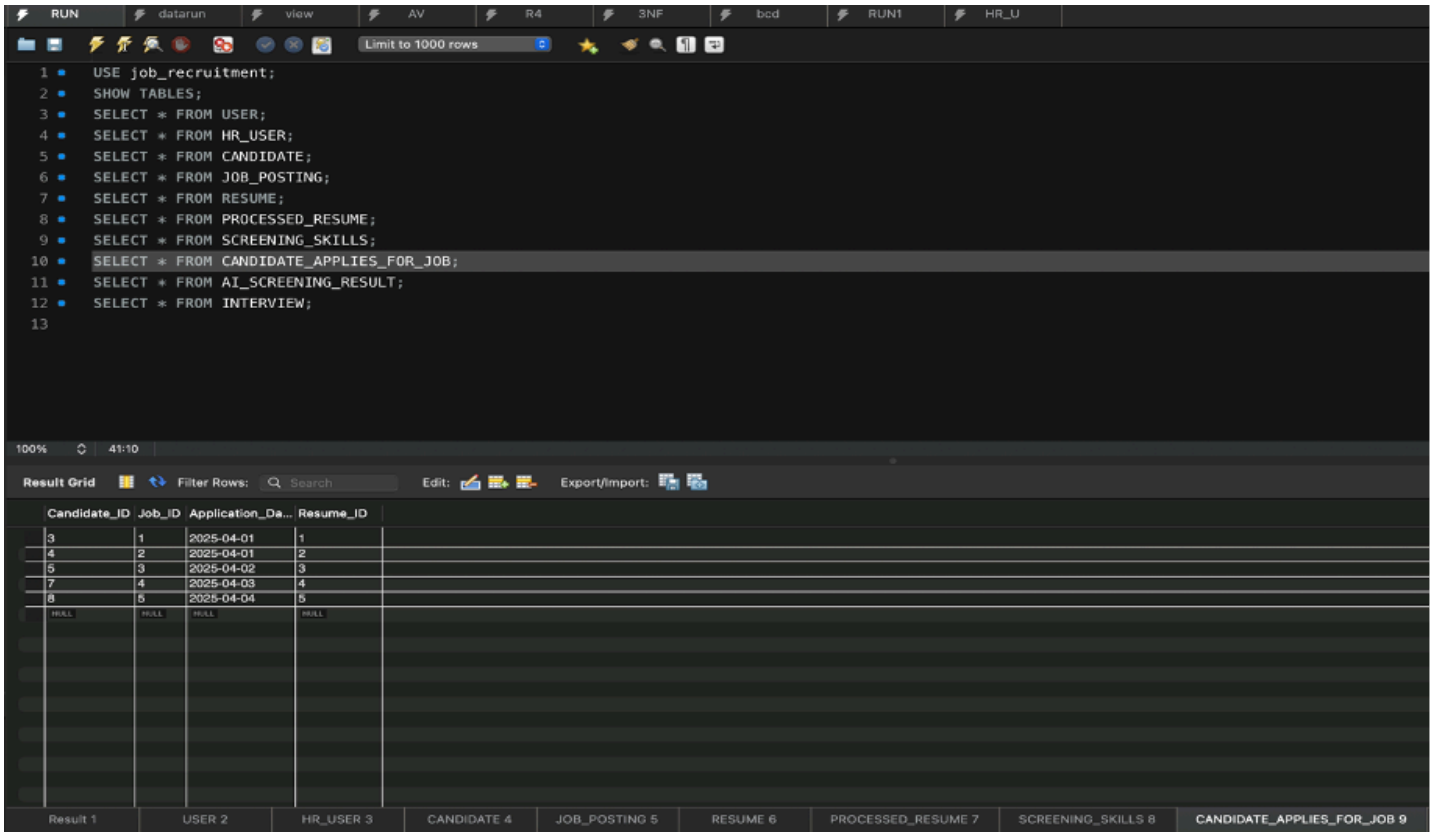
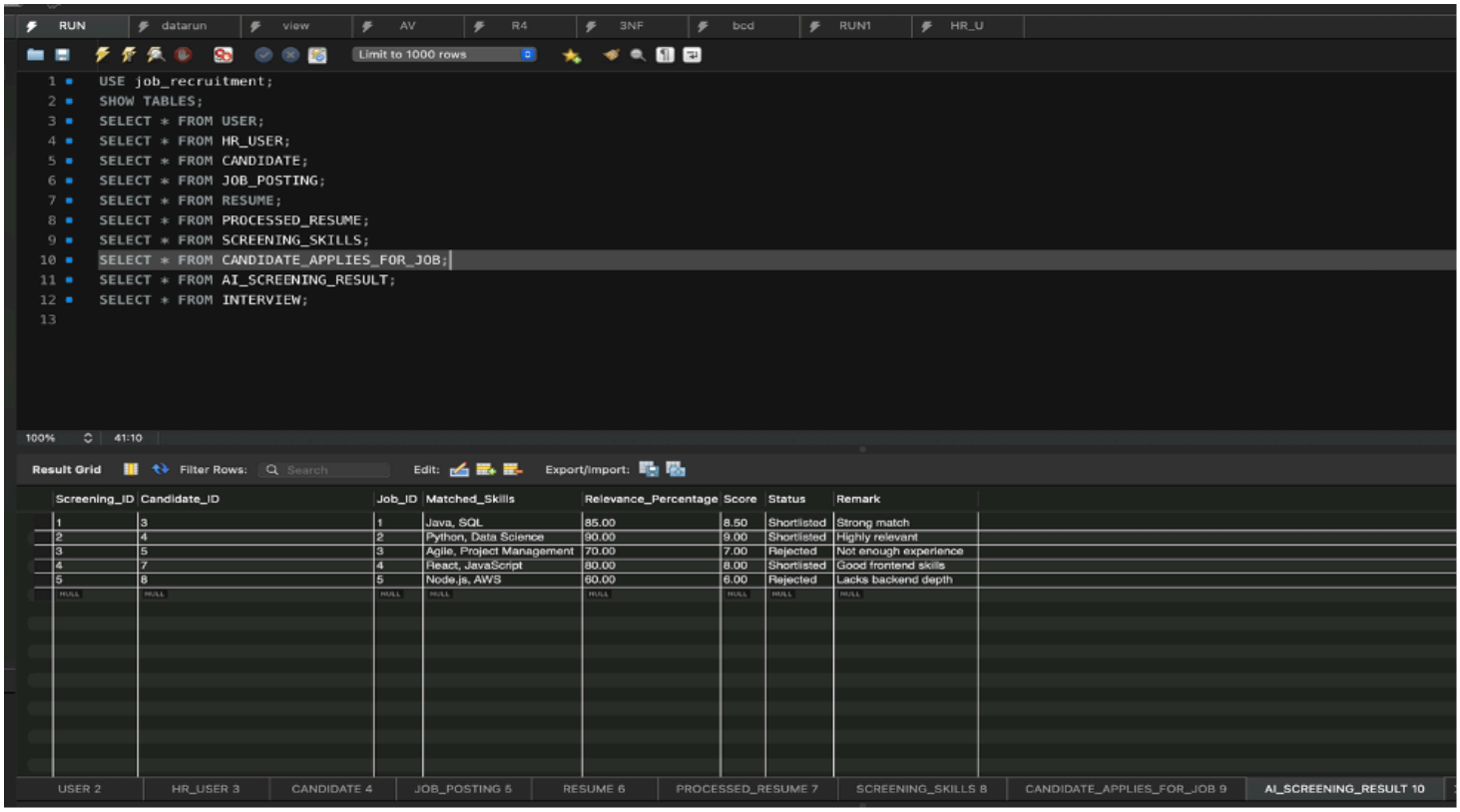


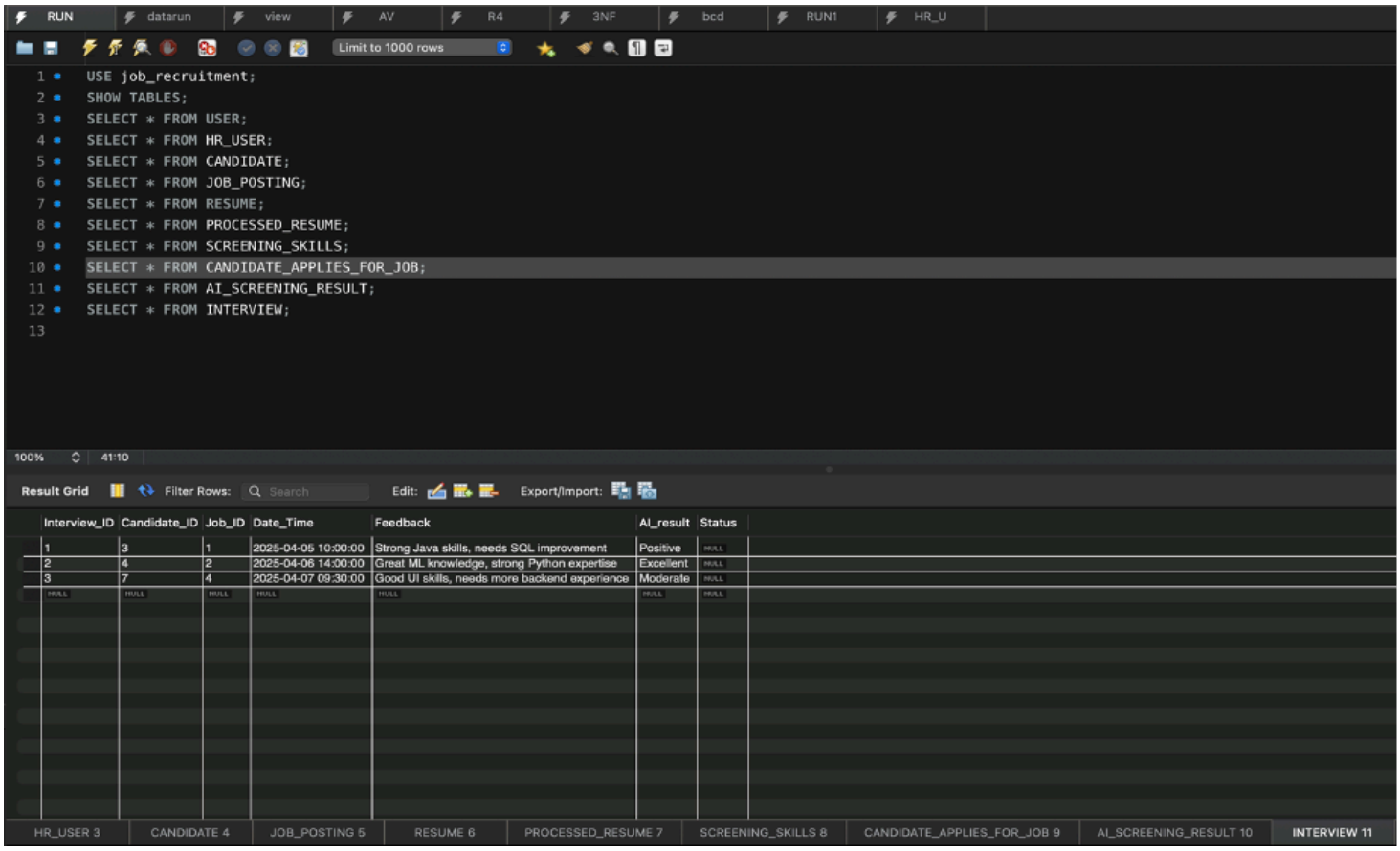
Fig 5 Creating SCREENING_SKILLS and CANDIDATE_APPLIES_FOR_JOB and inserting values



```
USE job_recruitment;
SHOW TABLES;
SELECT * FROM USER;
SELECT * FROM HR_USER;
SELECT * FROM CANDIDATE;
SELECT * FROM JOB_POSTING;
SELECT * FROM RESUME;
SELECT * FROM PROCESSED_RESUME;
SELECT * FROM SCREENING_SKILLS;
SELECT * FROM CANDIDATE_APPLIES_FOR_JOB;
SELECT * FROM AI_SCREENING_RESULT;
SELECT * FROM INTERVIEW;
```

Result Grid

Screening_ID	Candidate_ID	Job_ID	Matched_Skills	Relevance_Percentage	Score	Status	Remark
1	3	1	Java, SQL	85.00	8.50	Shortlisted	Strong match
2	4	2	Python, Data Science	90.00	9.00	Shortlisted	Highly relevant
3	5	3	Agile, Project Management	70.00	7.00	Rejected	Not enough experience
4	7	4	React, JavaScript	80.00	8.00	Shortlisted	Good frontend skills
5	8	5	Node.js, AWS	60.00	6.00	Rejected	Lacks backend depth

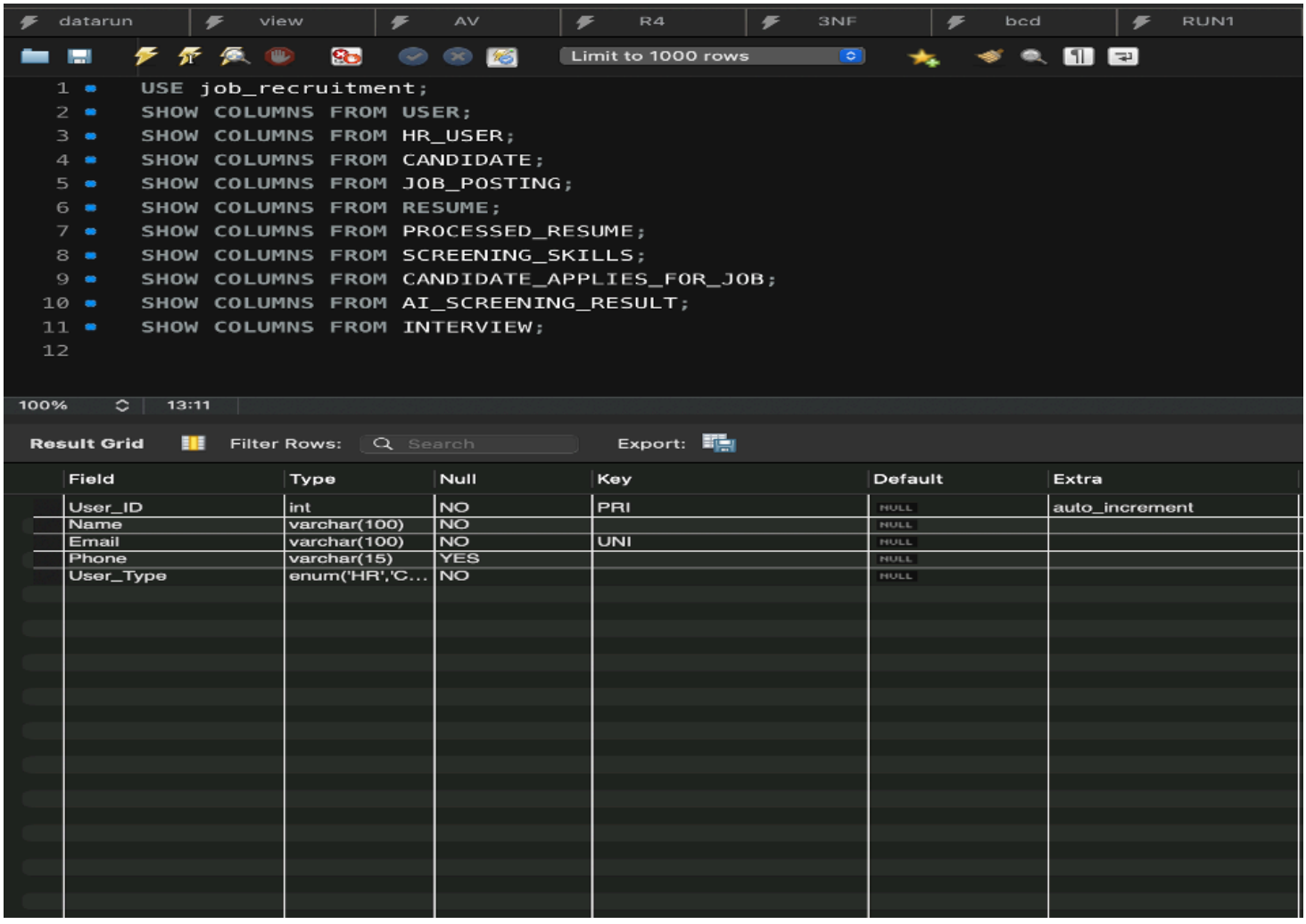


```
USE job_recruitment;
SHOW TABLES;
SELECT * FROM USER;
SELECT * FROM HR_USER;
SELECT * FROM CANDIDATE;
SELECT * FROM JOB_POSTING;
SELECT * FROM RESUME;
SELECT * FROM PROCESSED_RESUME;
SELECT * FROM SCREENING_SKILLS;
SELECT * FROM CANDIDATE_APPLIES_FOR_JOB;
SELECT * FROM AI_SCREENING_RESULT;
SELECT * FROM INTERVIEW;
```

Result Grid

Interview_ID	Candidate_ID	Job_ID	Date_Time	Feedback	AI_result	Status
1	3	1	2025-04-05 10:00:00	Strong Java skills, needs SQL improvement	Positive	
2	4	2	2025-04-06 14:00:00	Great ML knowledge, strong Python expertise	Excellent	
3	7	4	2025-04-07 09:30:00	Good UI skills, needs more backend experience	Moderate	

Fig 6 Creating AI_SCREENING_RESULT and INTERVIEW and inserting values



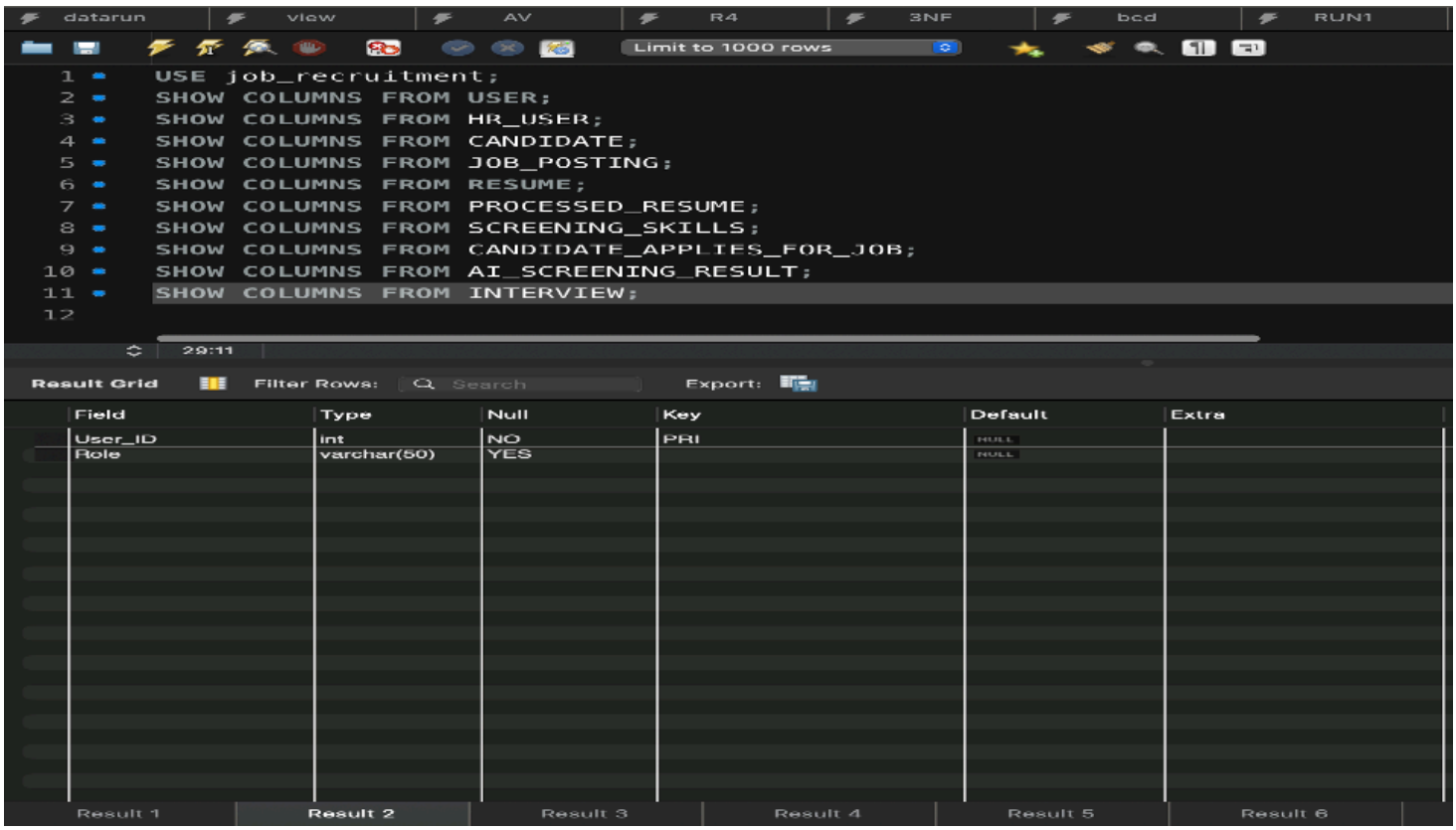
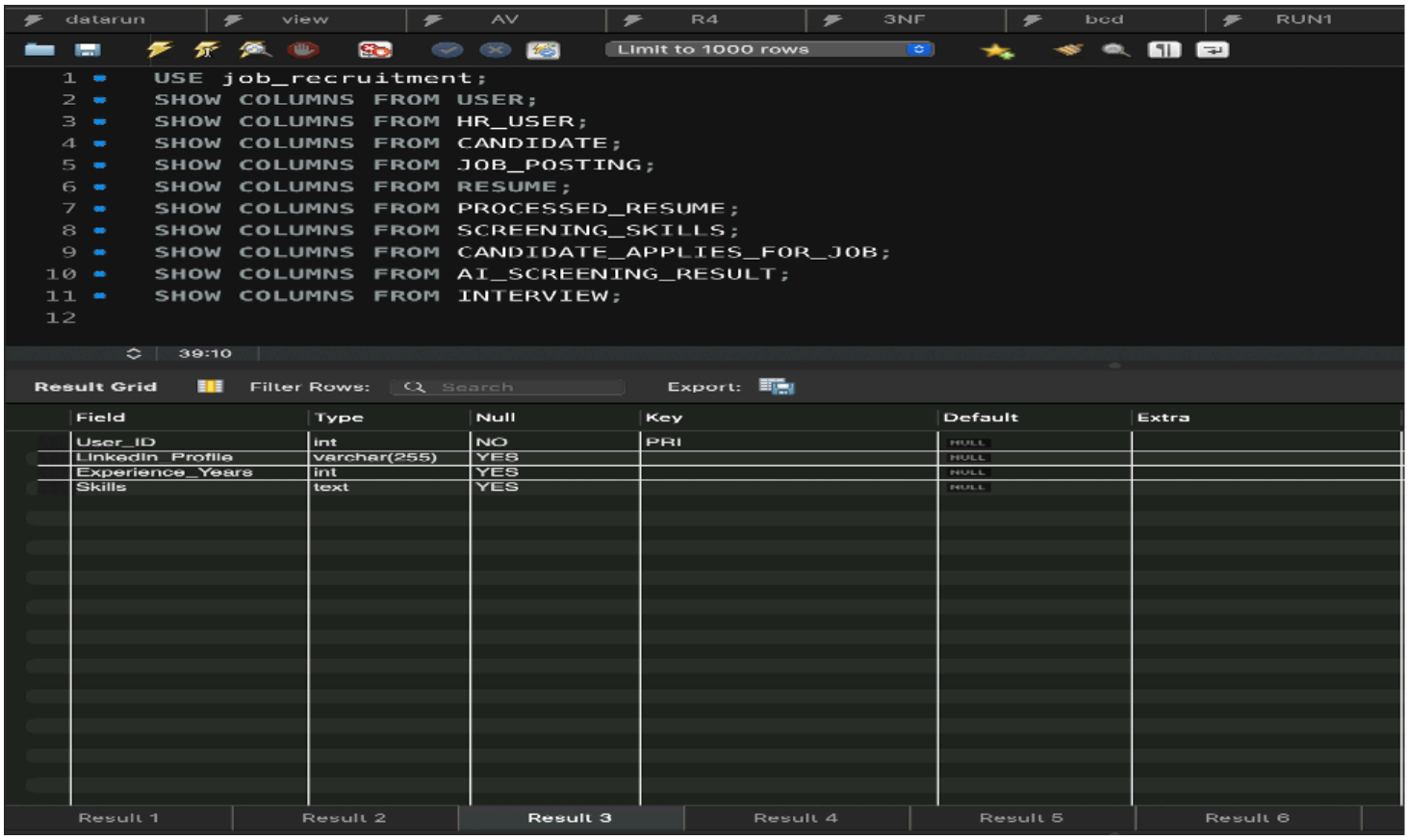


Fig 7 Displaying USER and HR_USER table



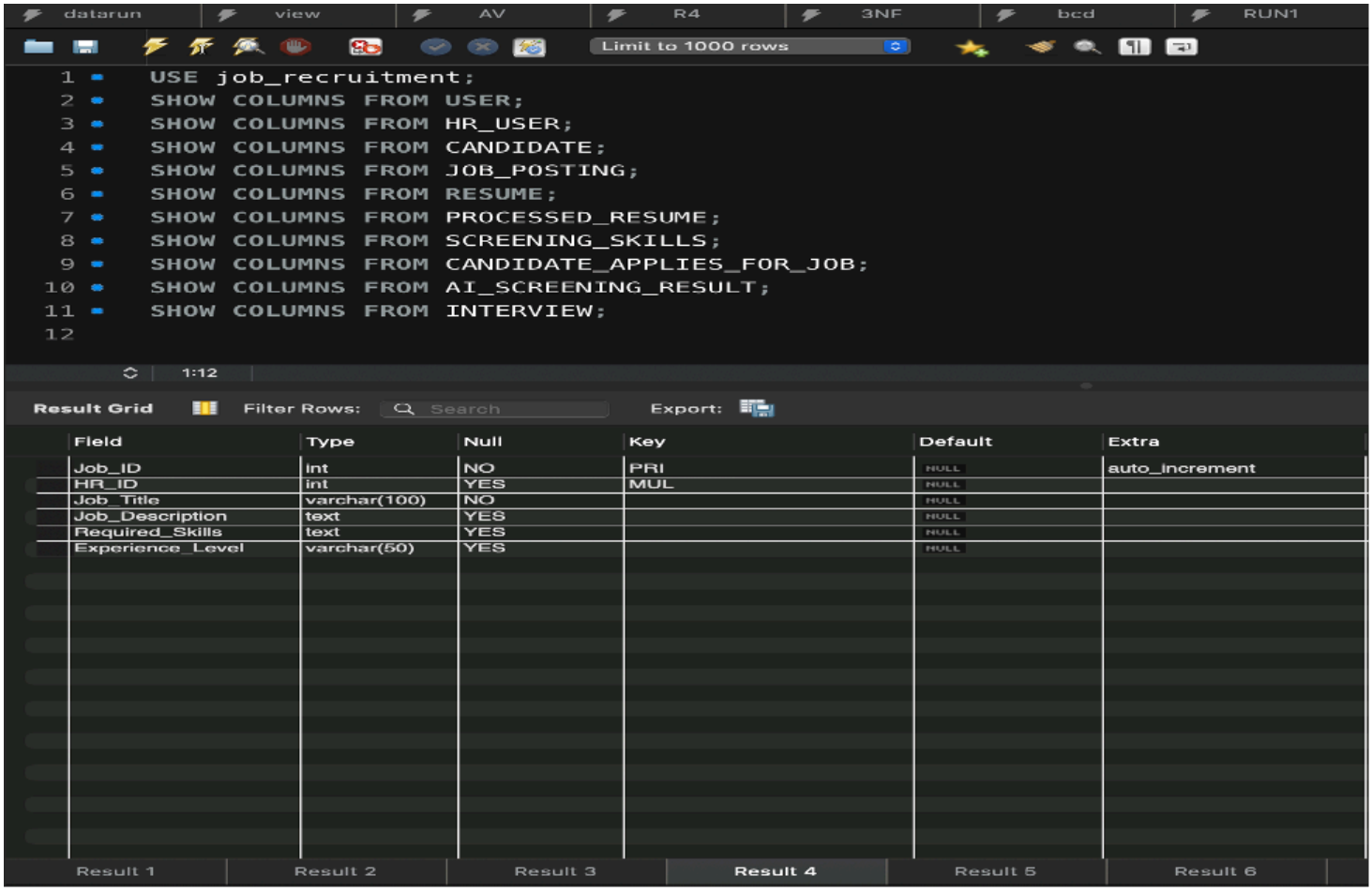
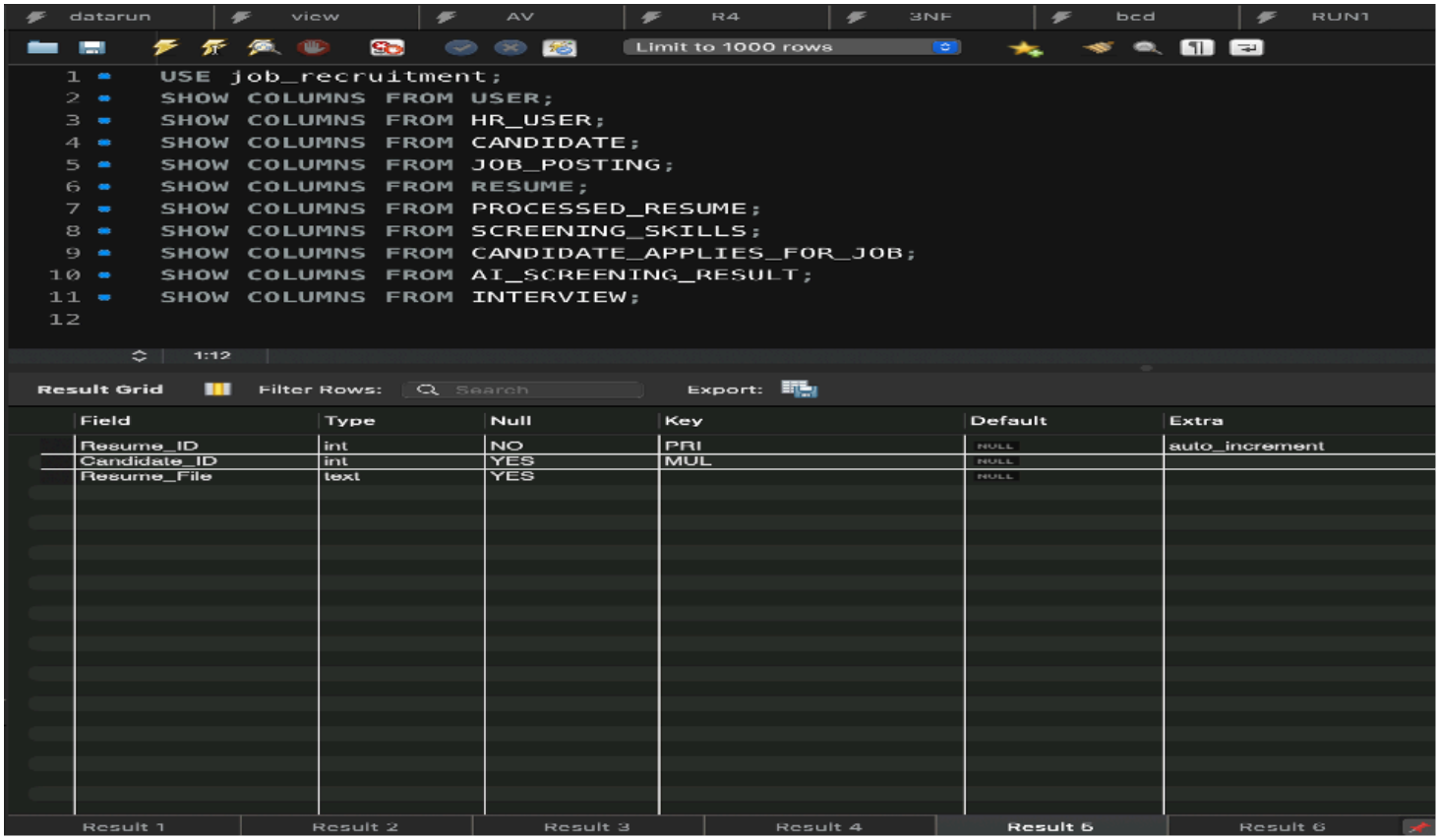


Fig 8 Displaying CANDIDATE and JOB_POSTING table



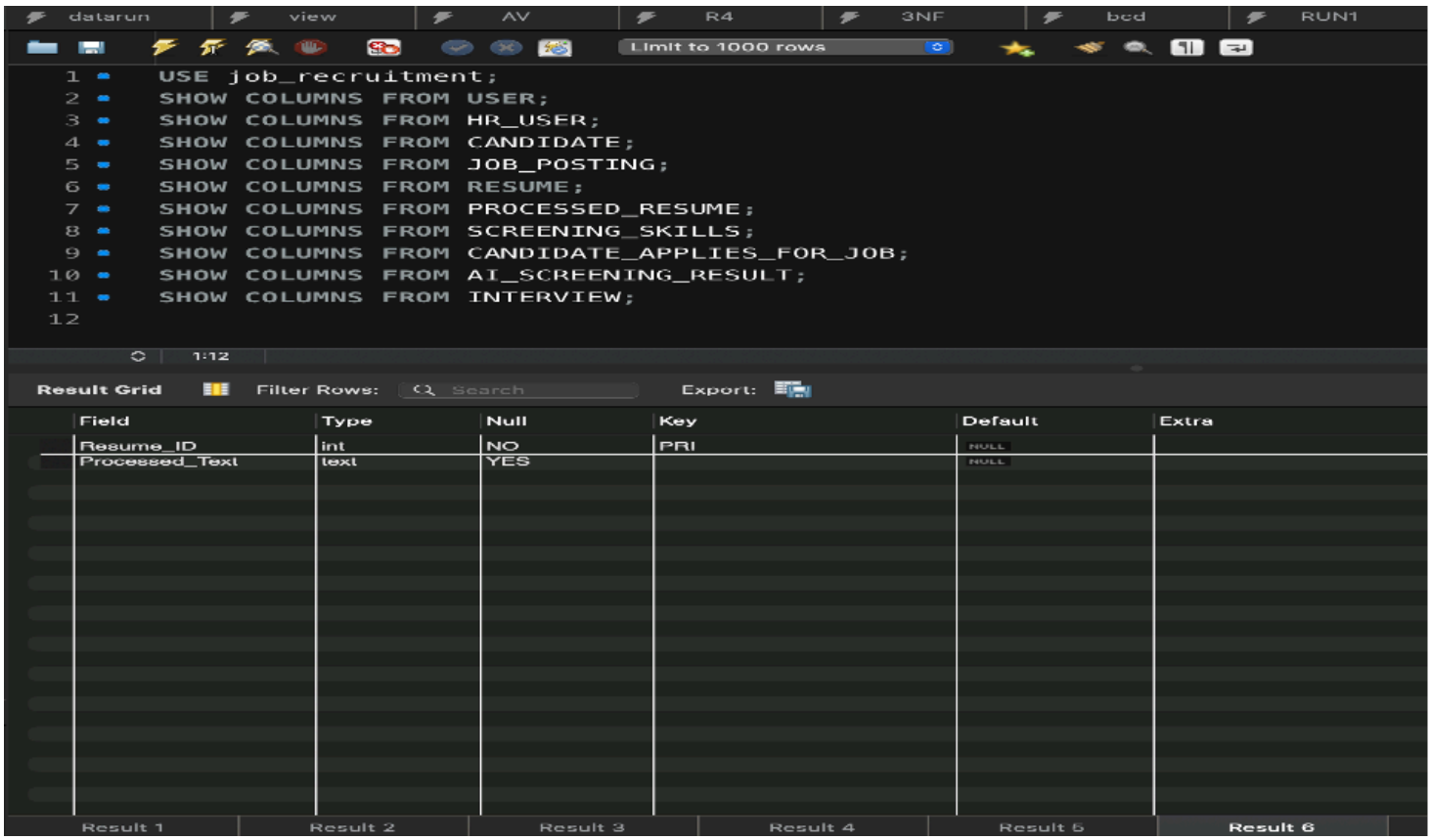
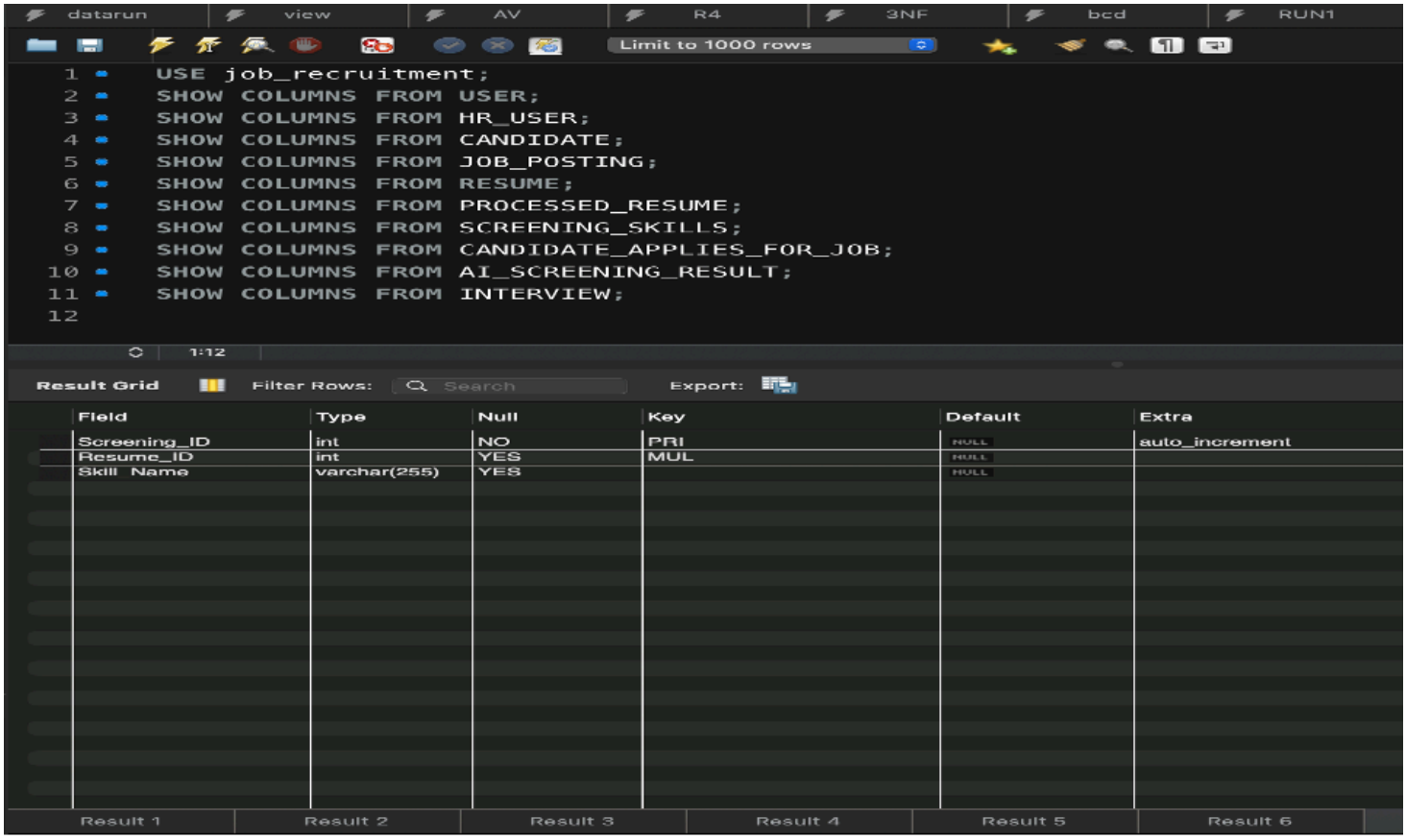


Fig 9 Displaying RESUME and PROCESSED_RESUME table



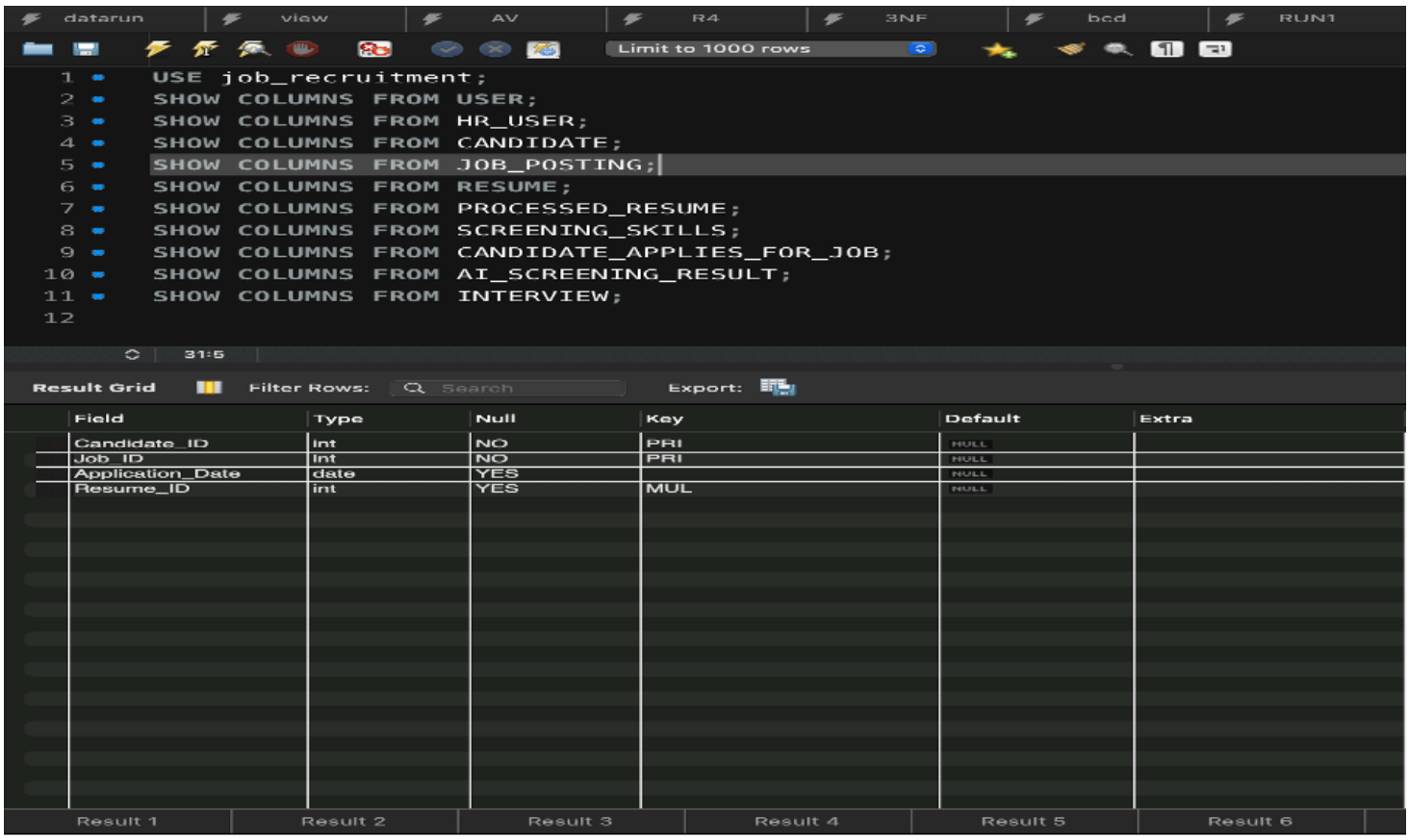
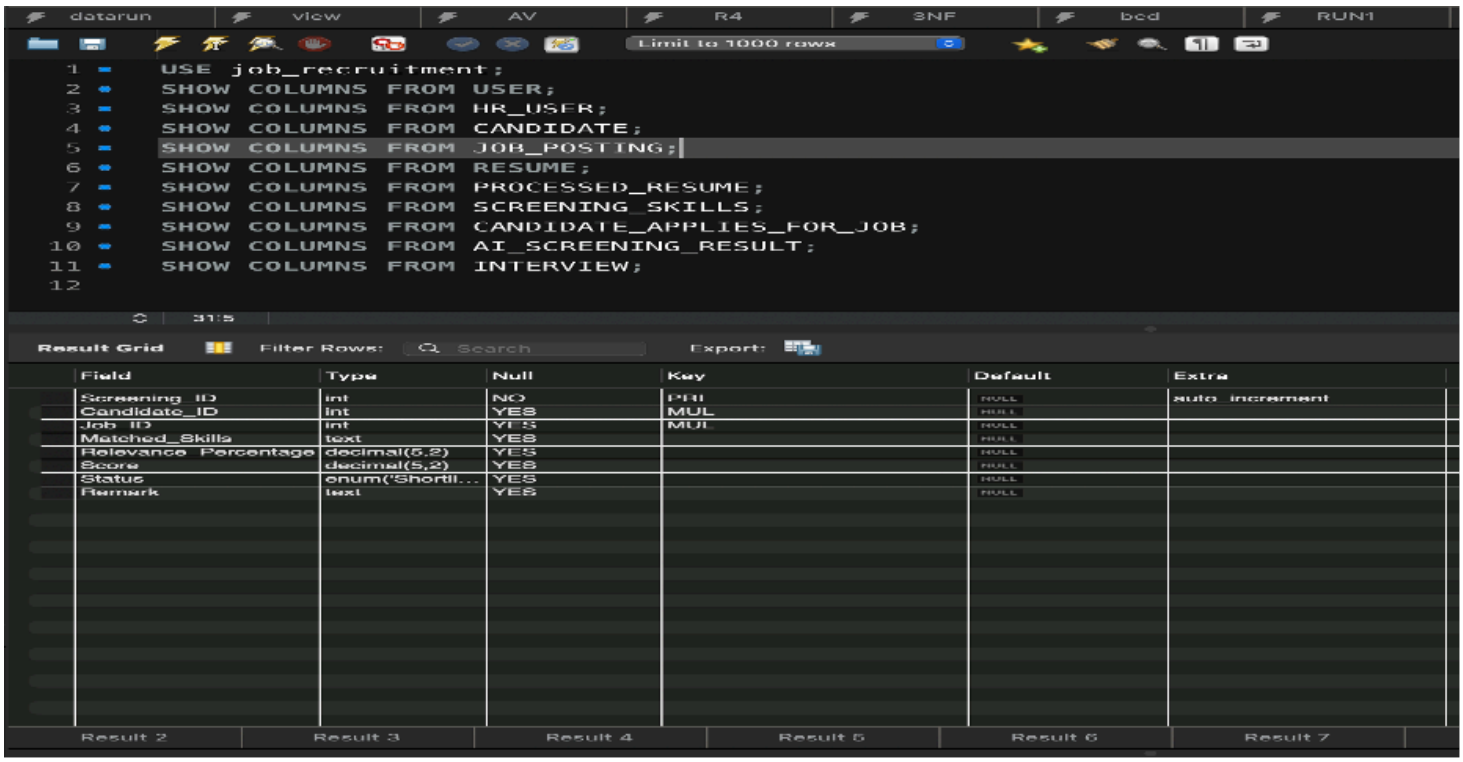


Fig 10 Displaying SCREENING_SKILLS and CANDIDATE_APPLIES_FOR_JOB table



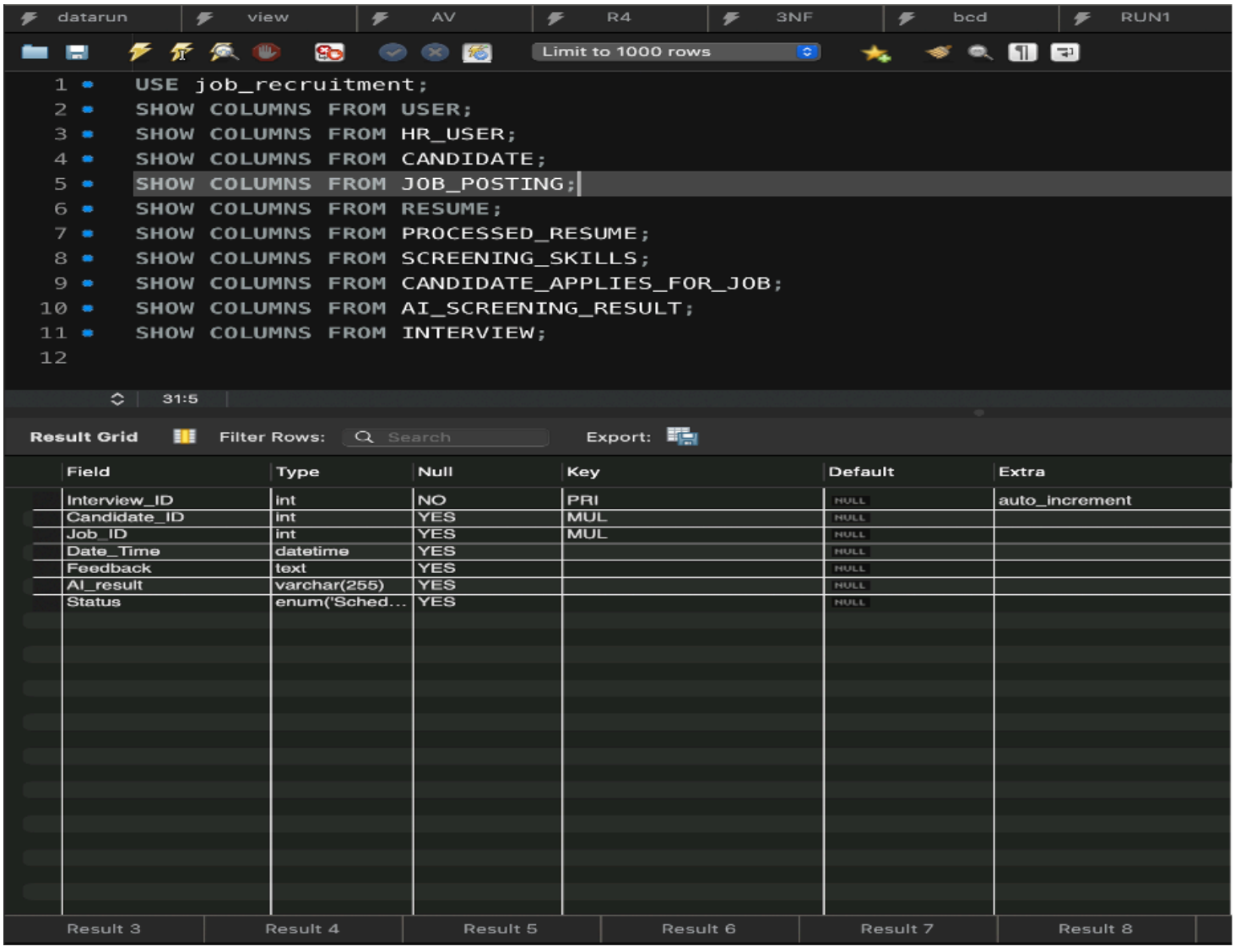


Fig 11 Displaying AI_SCREENING_RESULT and INTERVIEW table

DML commands

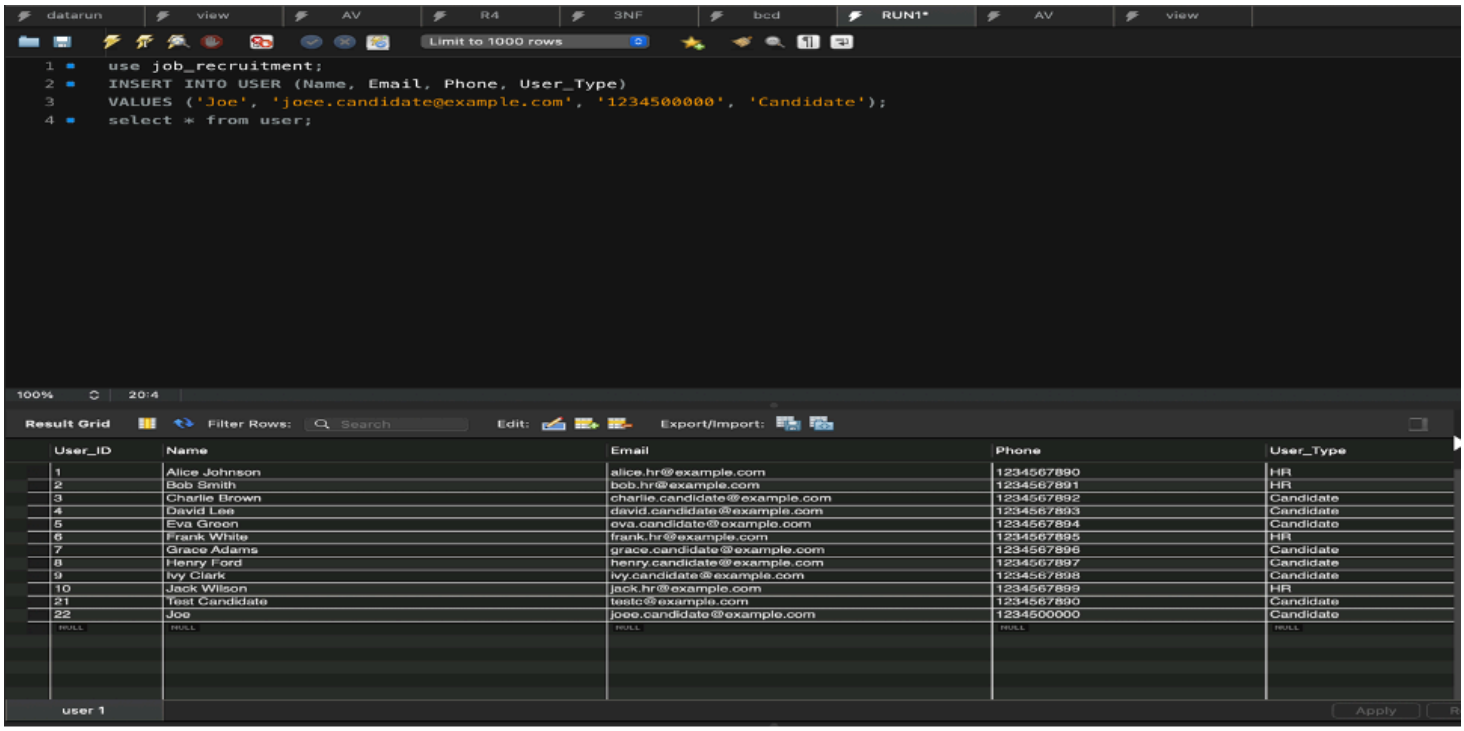


Fig 1 Inserting a new record into the table

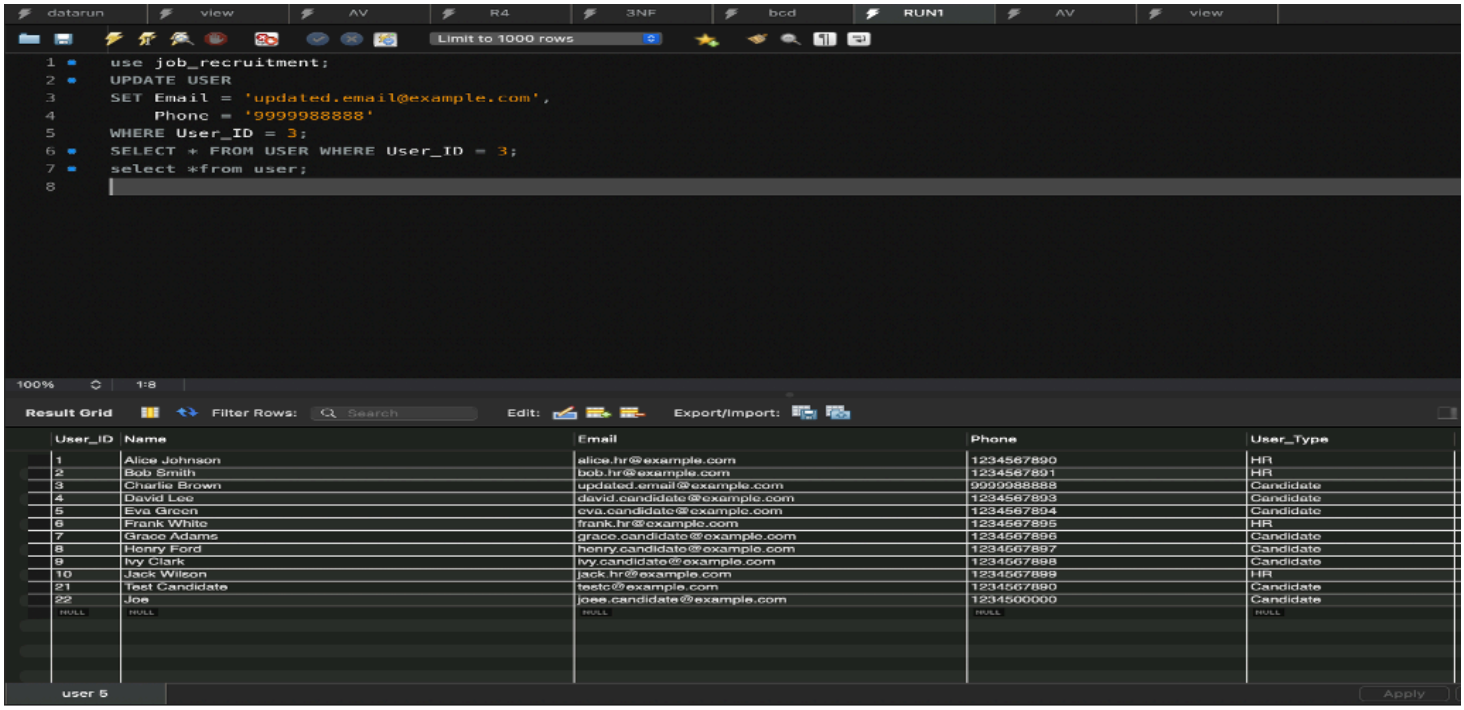


Fig 2 Updating a new record into the table

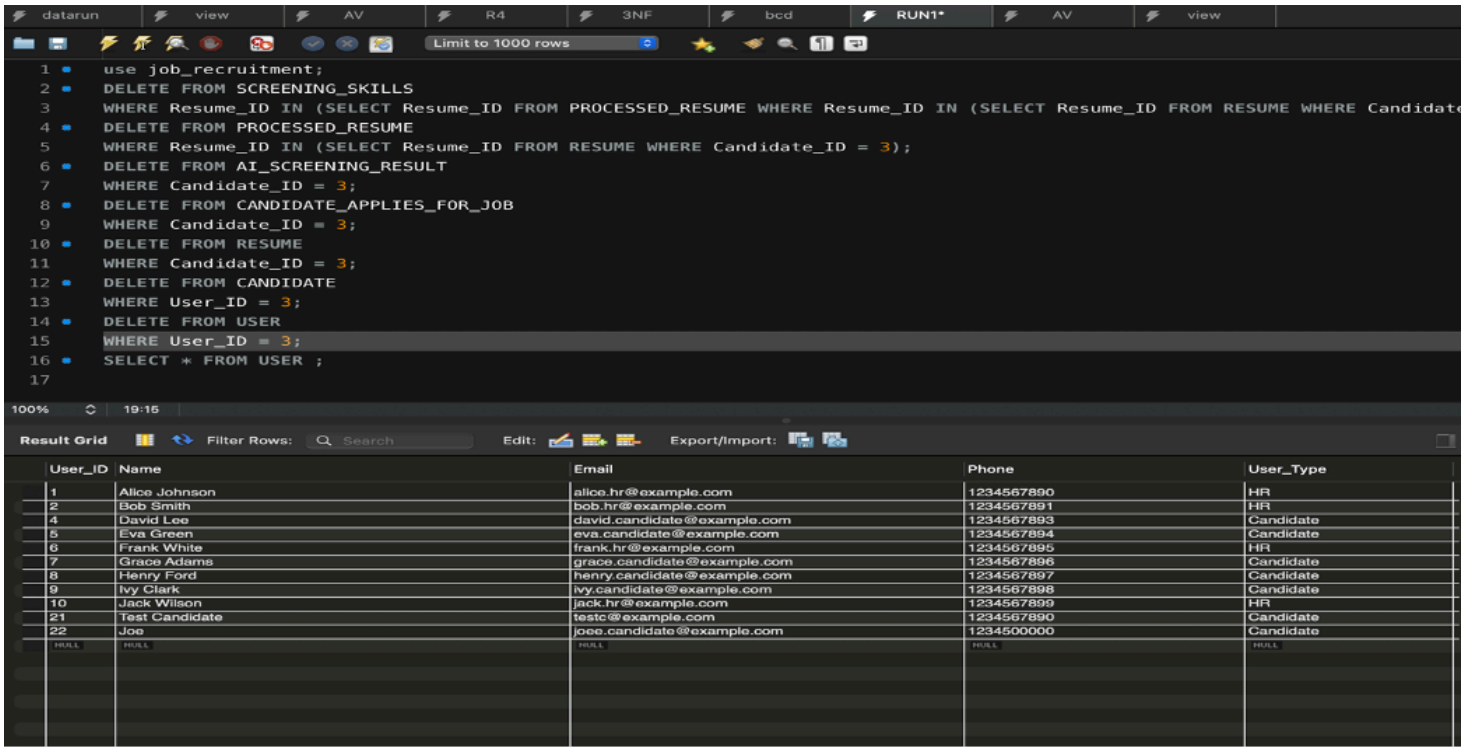


Fig 3 Deleting a record into the table

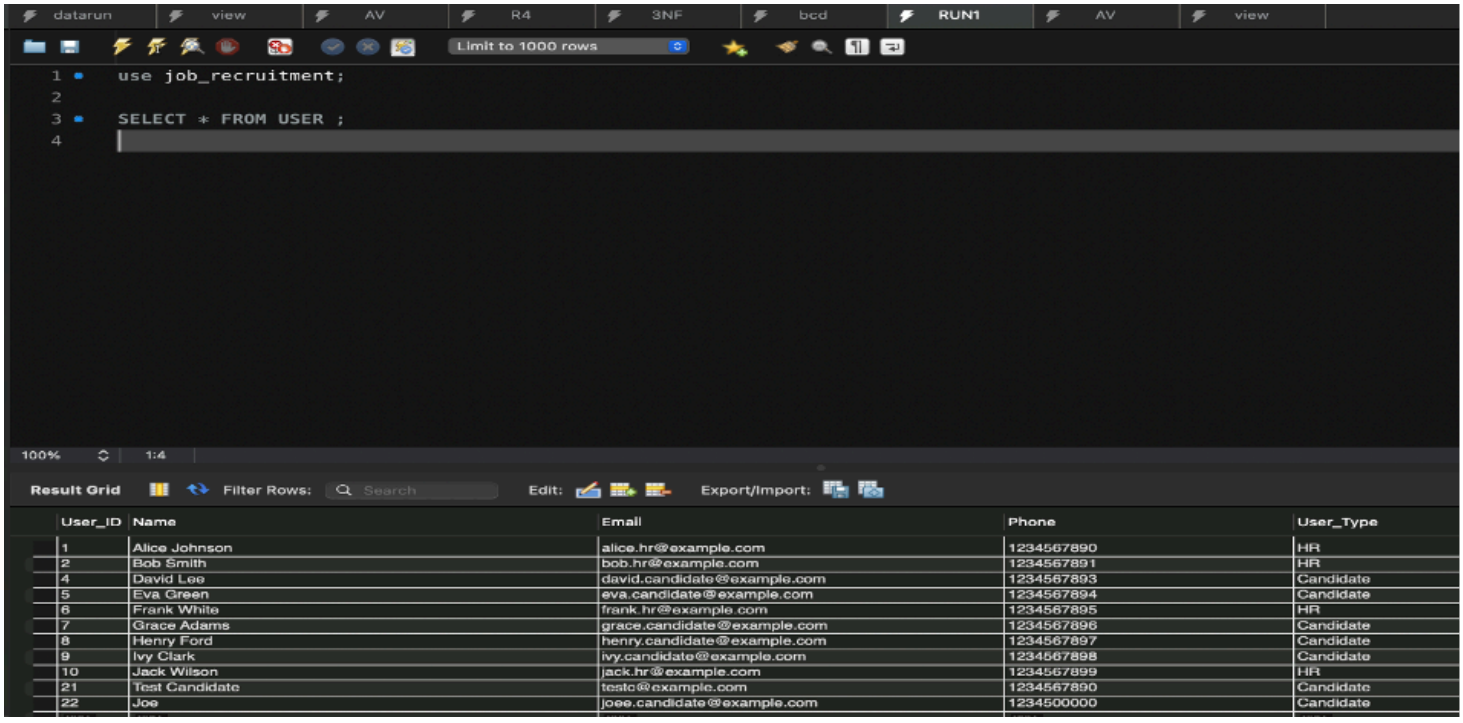


Fig 4 Selecting and viewing data from the table

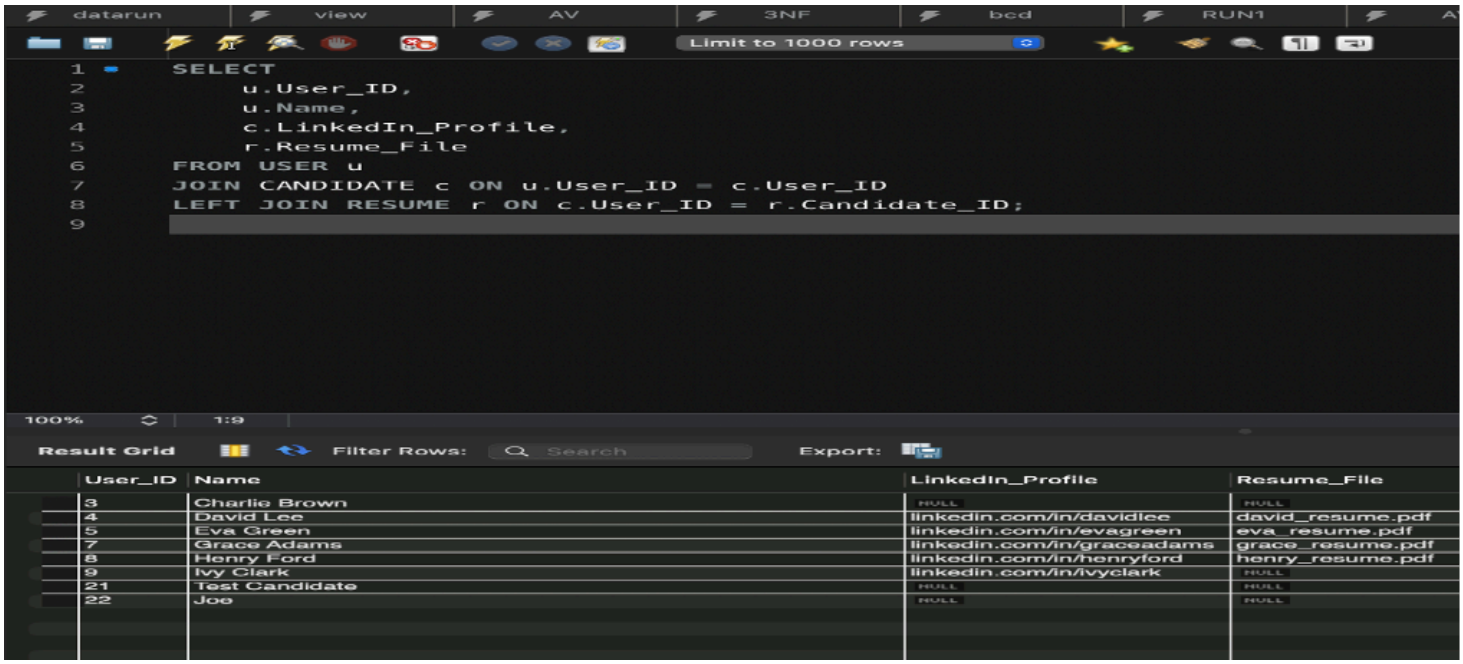


Fig 5 Left join

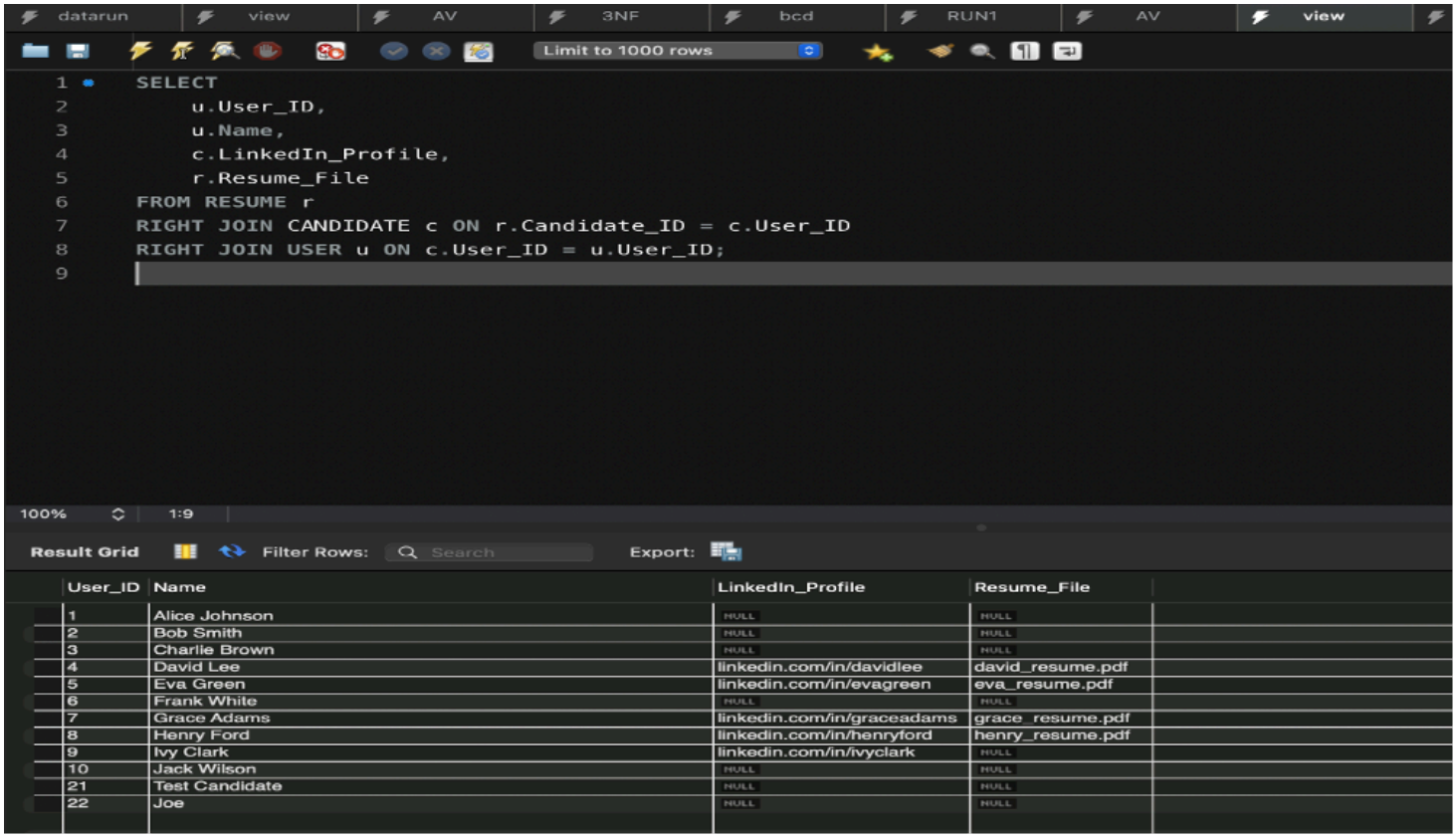


Fig 6 Right join

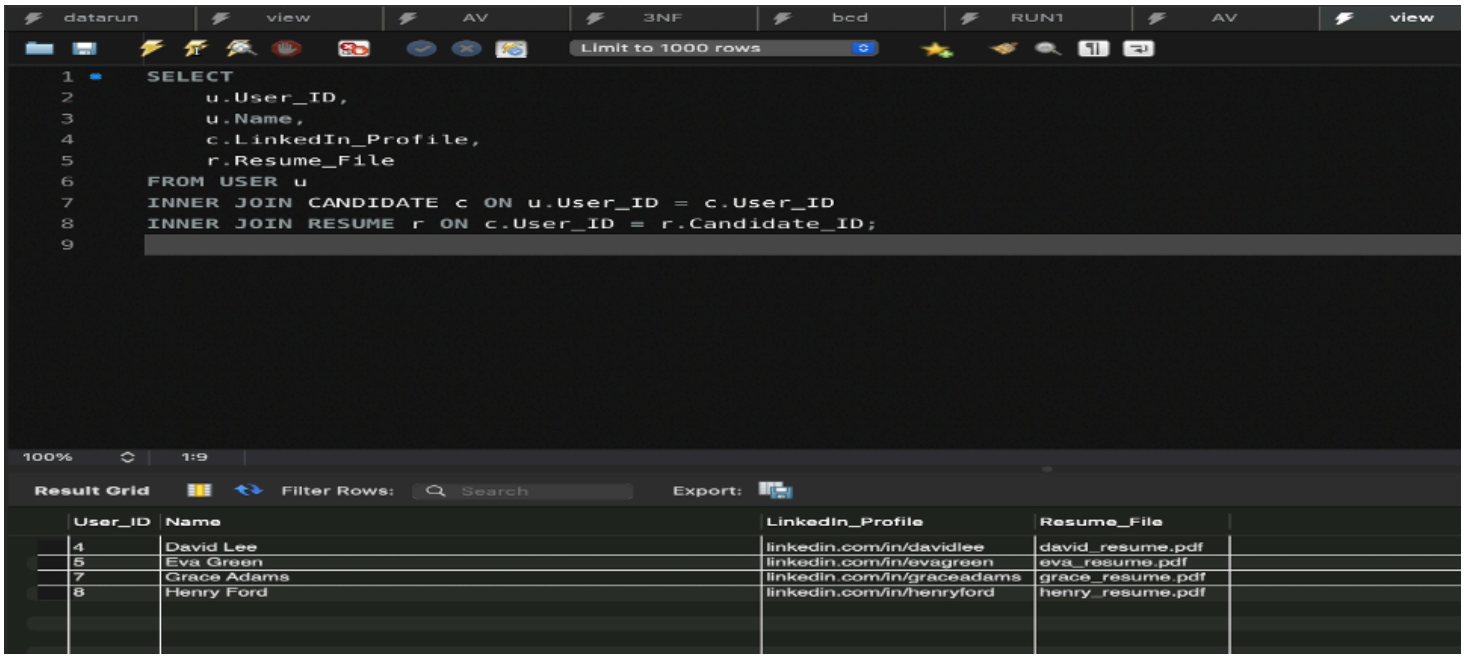


Fig 7 Inner join

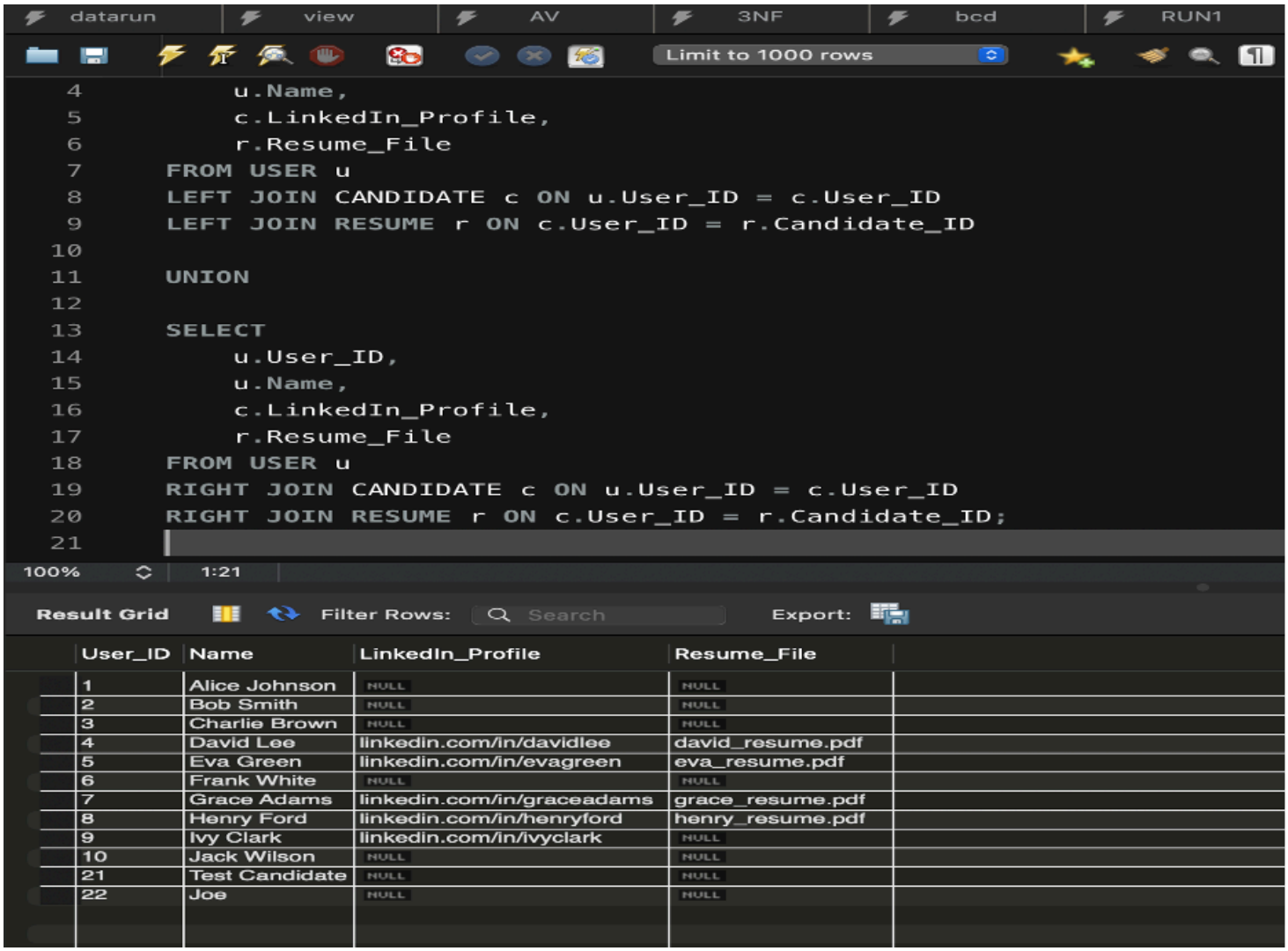


Fig 8 Outer join

COMPLEX QUERIES

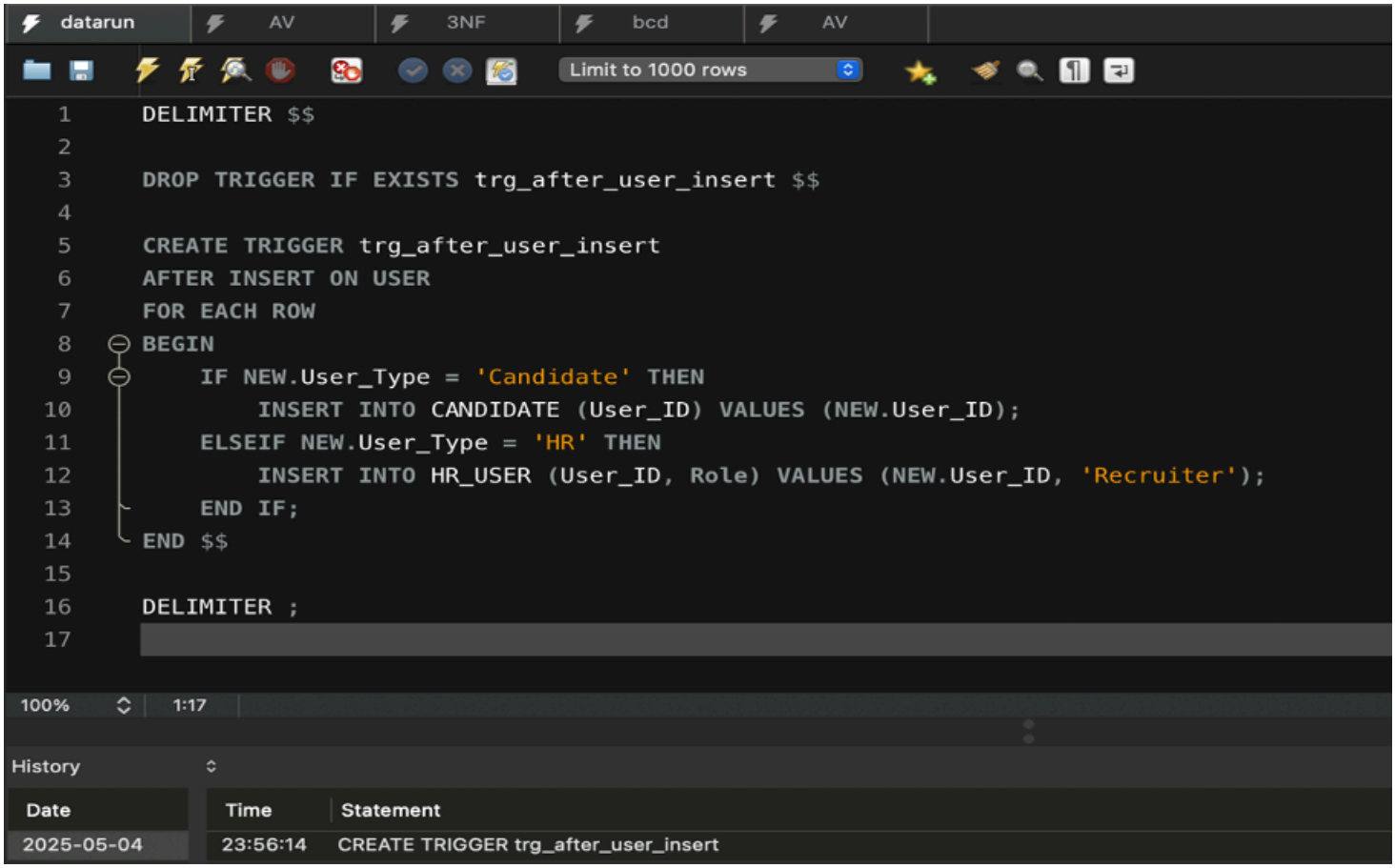


Fig 1 Trigger implementation

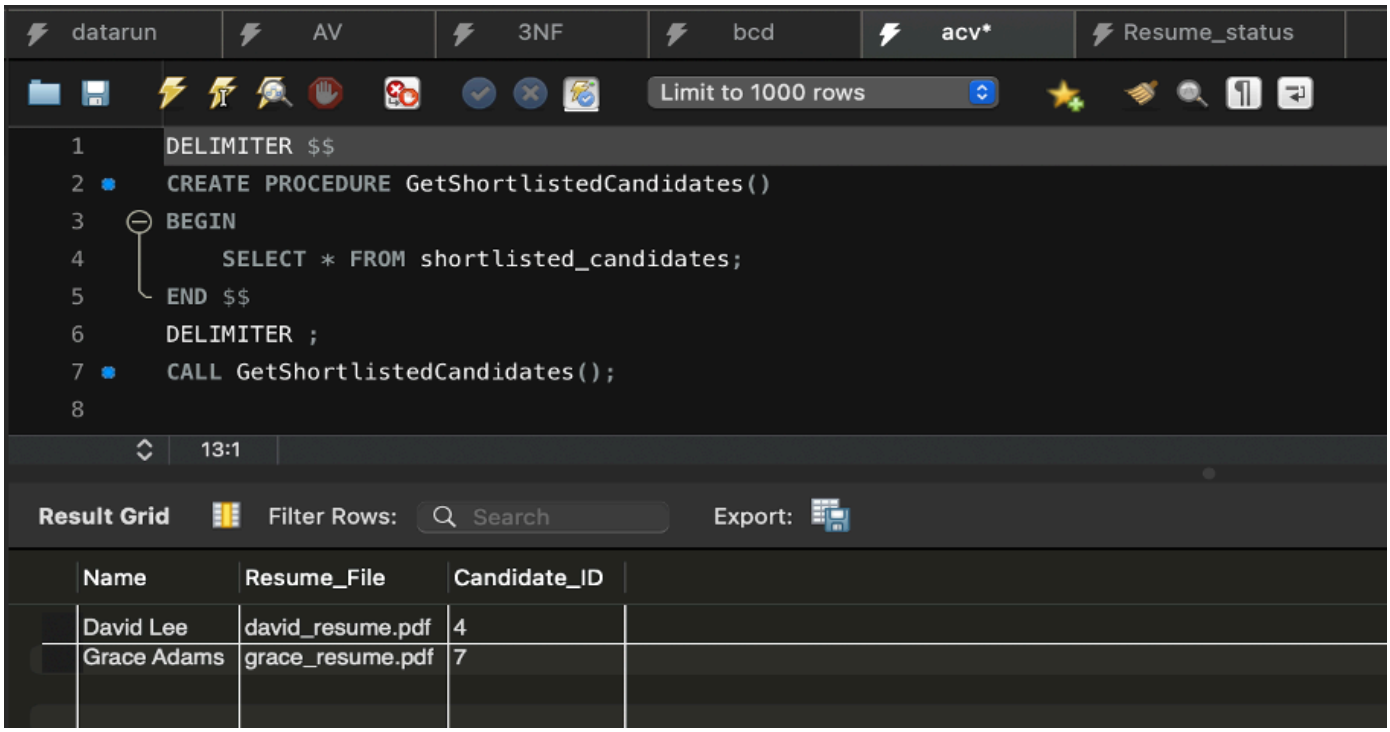


Fig 2 Stored-Procedure implementation

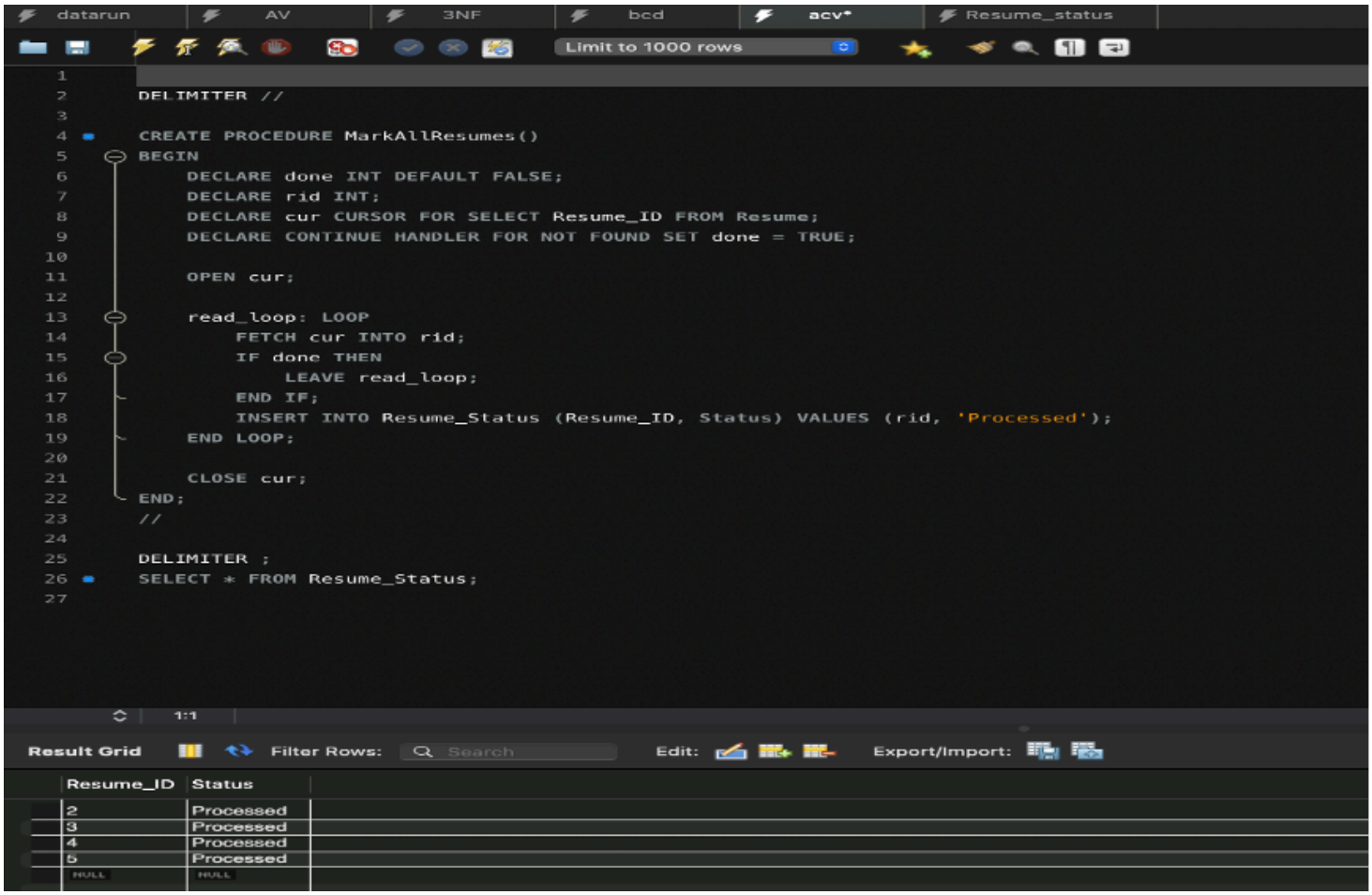


Fig 3 Cursor implementation

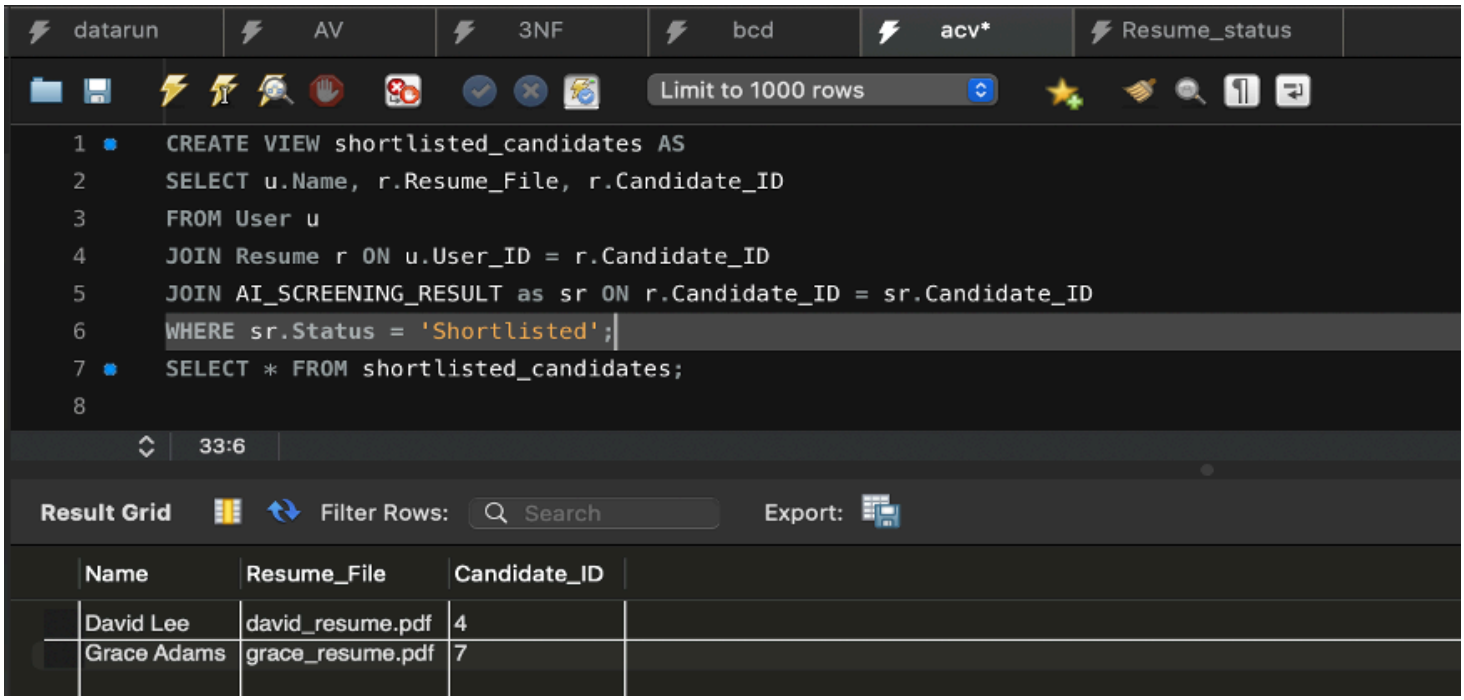
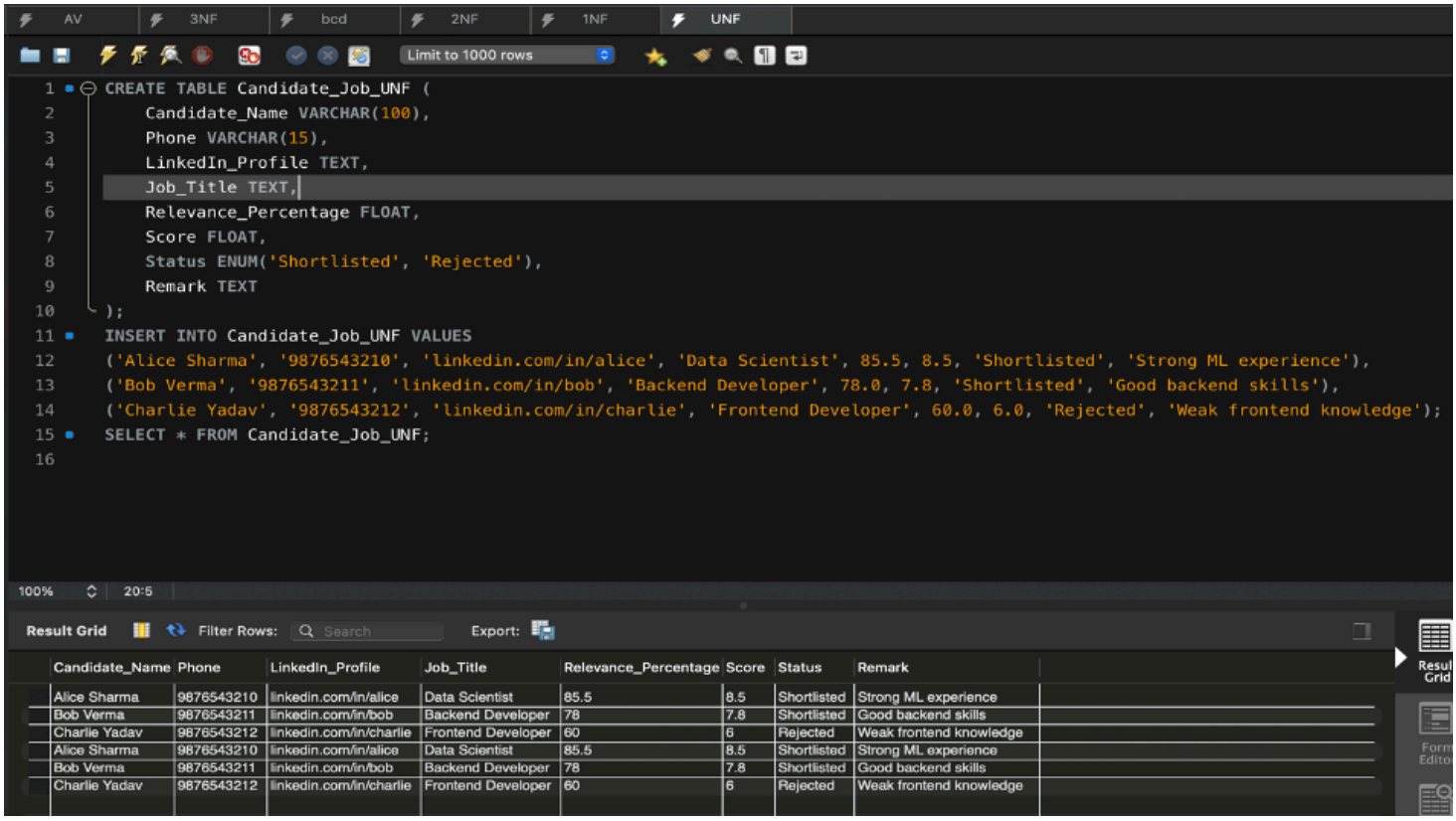


Fig 4 View implementation

NORMALIZATIONS

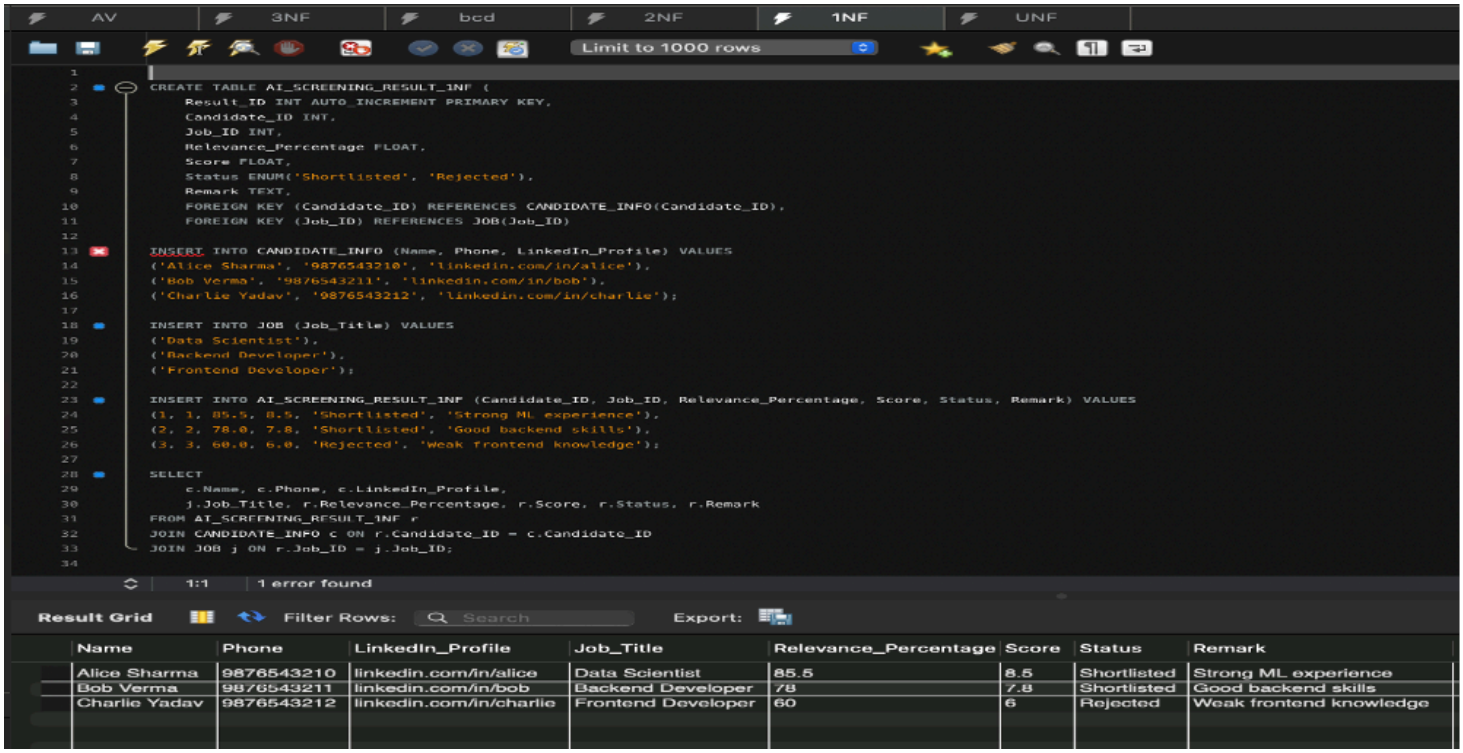


```
CREATE TABLE Candidate_Job_UNF (
    Candidate_Name VARCHAR(100),
    Phone VARCHAR(15),
    LinkedIn_Profile TEXT,
    Job_Title TEXT,
    Relevance_Percentage FLOAT,
    Score FLOAT,
    Status ENUM('Shortlisted', 'Rejected'),
    Remark TEXT
);
INSERT INTO Candidate_Job_UNF VALUES
('Alice Sharma', '9876543210', 'linkedin.com/in/alice', 'Data Scientist', 85.5, 8.5, 'Shortlisted', 'Strong ML experience'),
('Bob Verma', '9876543211', 'linkedin.com/in/bob', 'Backend Developer', 78.0, 7.8, 'Shortlisted', 'Good backend skills'),
('Charlie Yadav', '9876543212', 'linkedin.com/in/charlie', 'Frontend Developer', 60.0, 6.0, 'Rejected', 'Weak frontend knowledge');
SELECT * FROM Candidate_Job_UNF;
```

Result Grid

Candidate_Name	Phone	LinkedIn_Profile	Job_Title	Relevance_Percentage	Score	Status	Remark
Alice Sharma	9876543210	linkedin.com/in/alice	Data Scientist	85.5	8.5	Shortlisted	Strong ML experience
Bob Verma	9876543211	linkedin.com/in/bob	Backend Developer	78	7.8	Shortlisted	Good backend skills
Charlie Yadav	9876543212	linkedin.com/in/charlie	Frontend Developer	60	6	Rejected	Weak frontend knowledge
Alice Sharma	9876543210	linkedin.com/in/alice	Data Scientist	85.5	8.5	Shortlisted	Strong ML experience
Bob Verma	9876543211	linkedin.com/in/bob	Backend Developer	78	7.8	Shortlisted	Good backend skills
Charlie Yadav	9876543212	linkedin.com/in/charlie	Frontend Developer	60	6	Rejected	Weak frontend knowledge

Fig 1 Unnormalized Table



```
CREATE TABLE AI_SCREENING_RESULT_1NF (
    Result_ID INT AUTO_INCREMENT PRIMARY KEY,
    Candidate_ID INT,
    Job_ID INT,
    Relevance_Percentage FLOAT,
    Score FLOAT,
    Status ENUM('Shortlisted', 'Rejected'),
    Remark TEXT,
    FOREIGN KEY (Candidate_ID) REFERENCES CANDIDATE_INFO(Candidate_ID),
    FOREIGN KEY (Job_ID) REFERENCES JOB(Job_ID)

INSERT INTO CANDIDATE_INFO (Name, Phone, LinkedIn_Profile) VALUES
('Alice Sharma', '9876543210', 'linkedin.com/in/alice'),
('Bob Verma', '9876543211', 'linkedin.com/in/bob'),
('Charlie Yadav', '9876543212', 'linkedin.com/in/charlie');

INSERT INTO JOB (Job_Title) VALUES
('Data Scientist'),
('Backend Developer'),
('Frontend Developer');

INSERT INTO AI_SCREENING_RESULT_1NF (Candidate_ID, Job_ID, Relevance_Percentage, Score, Status, Remark) VALUES
(1, 1, 85.5, 8.5, 'Shortlisted', 'Strong ML experience'),
(2, 2, 78.0, 7.8, 'Shortlisted', 'Good backend skills'),
(3, 3, 60.0, 6.0, 'Rejected', 'Weak frontend knowledge');

SELECT
    c.Name, c.Phone, c.LinkedIn_Profile,
    j.Job_Title, r.Relevance_Percentage, r.Score, r.Status, r.Remark
FROM AI_SCREENING_RESULT_1NF r
JOIN CANDIDATE_INFO c ON r.Candidate_ID = c.Candidate_ID
JOIN JOB j ON r.Job_ID = j.Job_ID;
```

Result Grid

Name	Phone	LinkedIn_Profile	Job_Title	Relevance_Percentage	Score	Status	Remark
Alice Sharma	9876543210	linkedin.com/in/alice	Data Scientist	85.5	8.5	Shortlisted	Strong ML experience
Bob Verma	9876543211	linkedin.com/in/bob	Backend Developer	78	7.8	Shortlisted	Good backend skills
Charlie Yadav	9876543212	linkedin.com/in/charlie	Frontend Developer	60	6	Rejected	Weak frontend knowledge

Fig 2 1NF Table

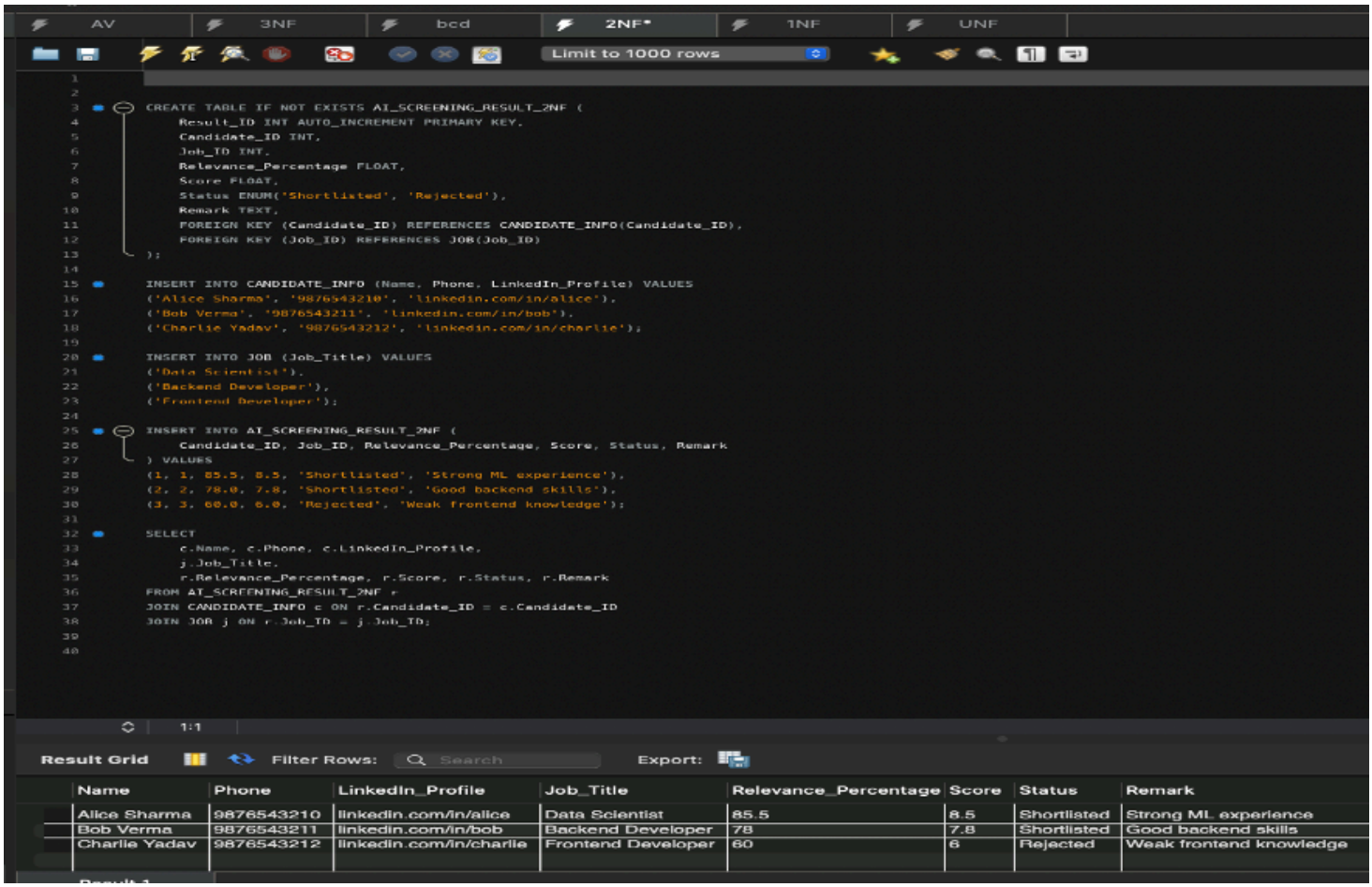


Fig 3 2NF Table

Fig 4 3NF Table

CONCURRENCY CONTROL

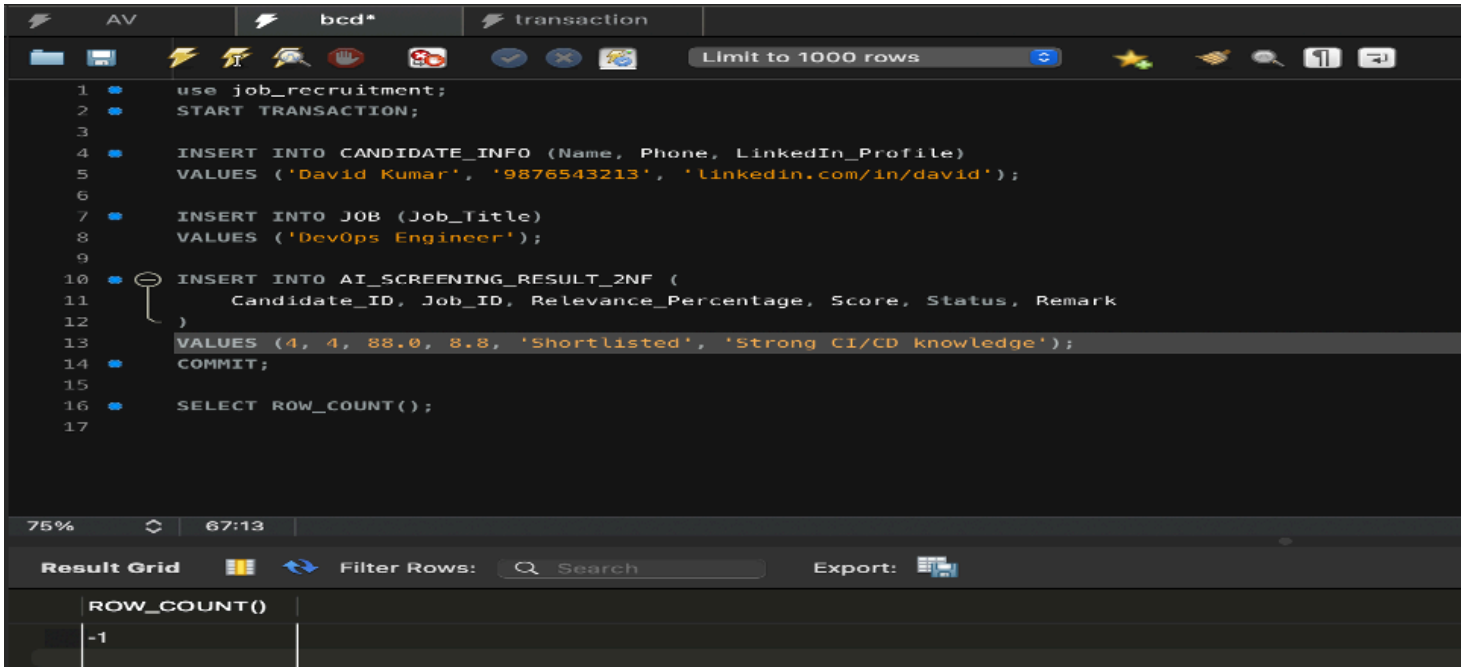


Fig 1 Transaction Handling

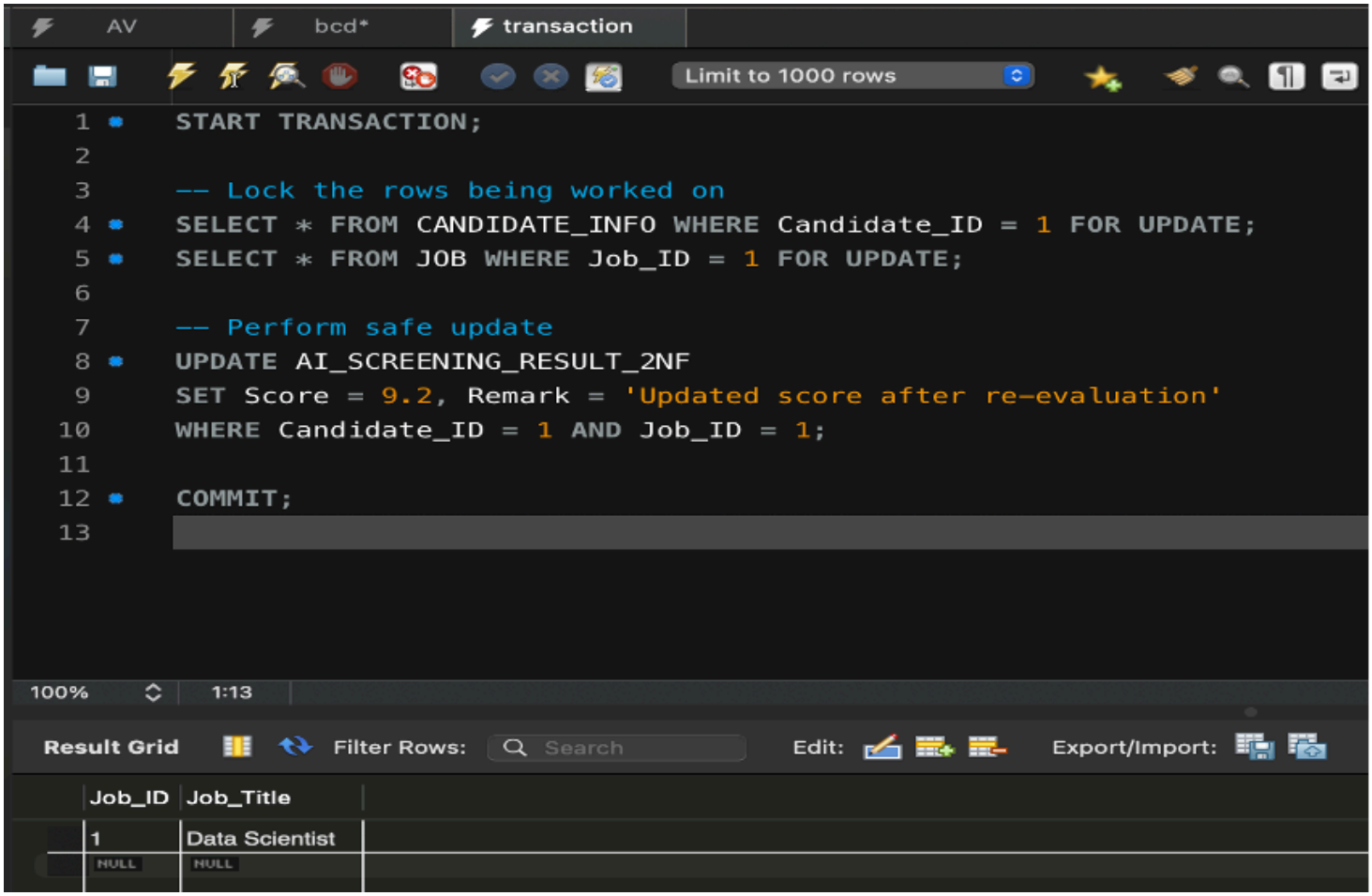


Fig 2 Concurrency Control

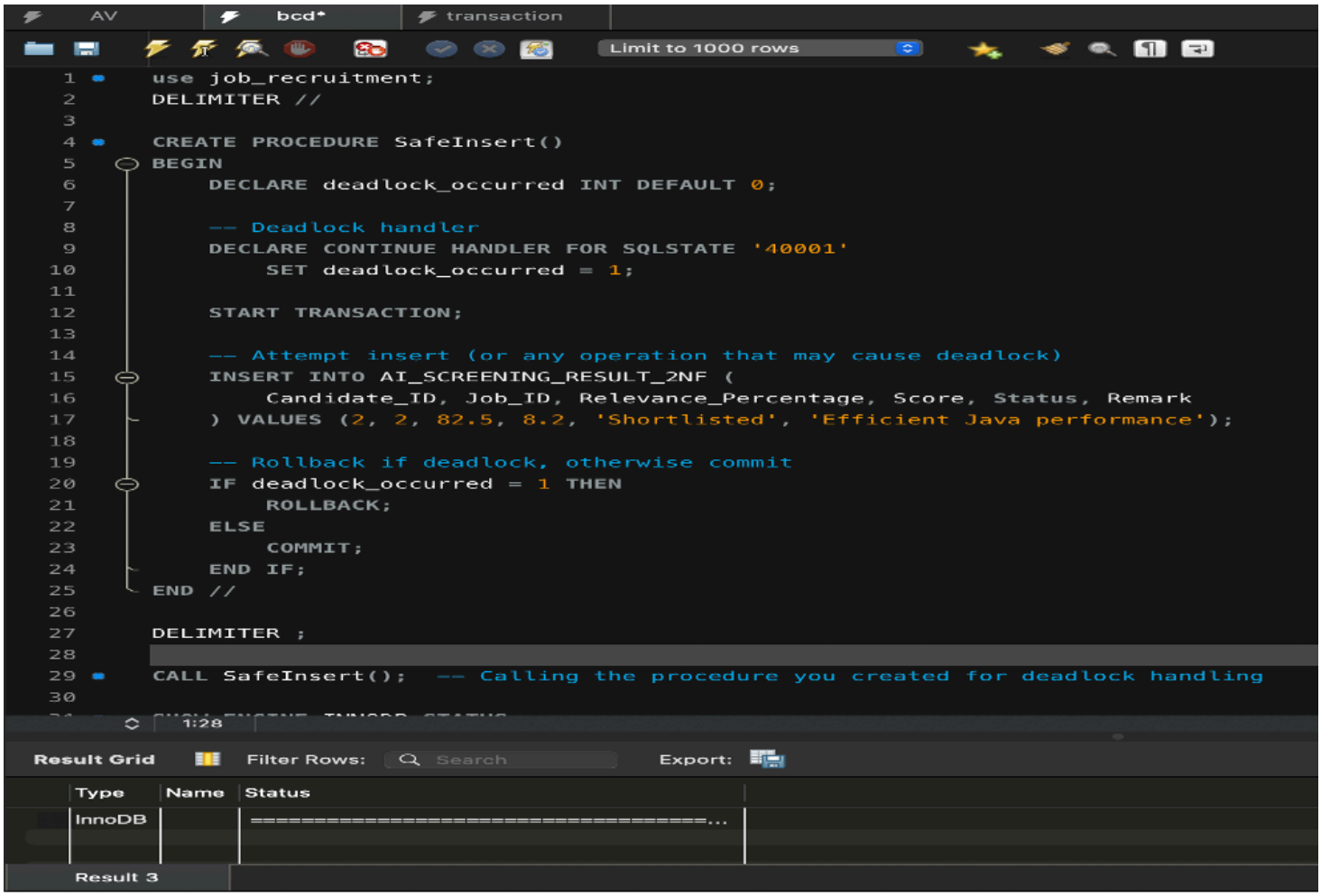


Fig 3 Deadlock Handling