Lecture Notes – Introduction To Sub queries

Section 1: Lecture Summary

Subqueries are queries nested inside other queries, also known as inner queries or nested queries. The inner query executes first, and its result is used by the outer query to simplify complex logic. Subqueries can return a single **scalar value** (e.g., from aggregate functions like average), a list of values, or a table, and they are used in SELECT, FROM, WHERE, HAVING clauses, as well as INSERT, DELETE, and UPDATE statements.

Section 2: Key Concepts and Explanations

Outer query is the main query that uses the subquery result. **Inner query** (subquery) runs first and provides data for comparison or joining. Subqueries simplify complex conditions by breaking them into parts. In **WHERE** clause, subqueries support scalar or list results for operators like =, >, IN. In **SELECT**, they must return a scalar. In **FROM**, they return a table for joining. In **HAVING**, scalar only. Example logic: to find employees with salary higher than company average, first compute the average (scalar subquery), then filter in the outer query.

Section 3: Example Code and Use Cases

```
SELECT *
FROM Employees
WHERE Salary > (SELECT AVG(Salary) FROM Employees);
```

This query finds all employees whose salary exceeds the average salary across all employees. The subquery `(SELECT AVG(Salary) FROM Employees)` returns a single scalar value (average salary, e.g., 70600), which the outer query uses in the WHERE condition to filter results from the **Employees** table.

Section 4: Key Takeaways

Subqueries nest inside clauses like SELECT, FROM, WHERE, HAVING for scalar, list, or table results. Inner query executes first, feeding results to outer query. Use for complex logic like comparing to aggregates (e.g., salary > average). Applicable in DML operations like INSERT, DELETE, UPDATE.