Lecture Notes – Limit & Offset

Section 1: Lecture Summary

Limit restricts the number of rows returned in a query result, while **offset** skips a specified number of initial rows before applying the limit. These clauses are used after **order by** to control result pagination, such as showing rows 6 to 10 by offsetting 5 rows and limiting to 5 rows from the Employees table.

Section 2: Key Concepts and Explanations

Limit specifies the maximum number of rows to return from the query result, for example, limiting 10 rows to 5. **Offset** skips the first N rows of the result set, allowing retrieval of later portions like rows starting from the 6th row. These clauses require **order by** for consistent results, as they operate on the sorted output. For descending salary order, top earners appear first, so offset skips them to show subsequent rows. For dates, ascending **hireDate** order places oldest hires (senior most) first, as earlier dates have smaller values.

Section 3: Example Code and Use Cases

Query to show employees in rows 6 to 10 from Employees table (skip 5, limit 5):

```
SELECT * FROM Employees LIMIT 5 OFFSET 5;
```

This returns rows 6 through 10 after any implicit ordering.

Query to list **EmpID** and **Salary** of 5 employees after ignoring top 2 earners:

```
SELECT EmpID, Salary FROM Employees ORDER BY Salary DESC LIMIT 5 OFFSET 2;
```

Sorting **Salary** descending places highest first; offset 2 skips them.

Query to find 3 senior most employees (oldest **HireDate**):

```
SELECT EmpID, FirstName, LastName, HireDate FROM Employees ORDER BY
HireDate ASC LIMIT 3;
```

Ascending **HireDate** brings earliest dates first; limit 3 gets top 3.

Section 4: Key Takeaways

Use **limit** to cap row count and **offset** to skip initial rows after **order by**. Combine with descending sorts for top-N exclusions (e.g., salaries) or ascending for earliest values (e.g., dates). Practice on companyDB Employees table for pagination scenarios.