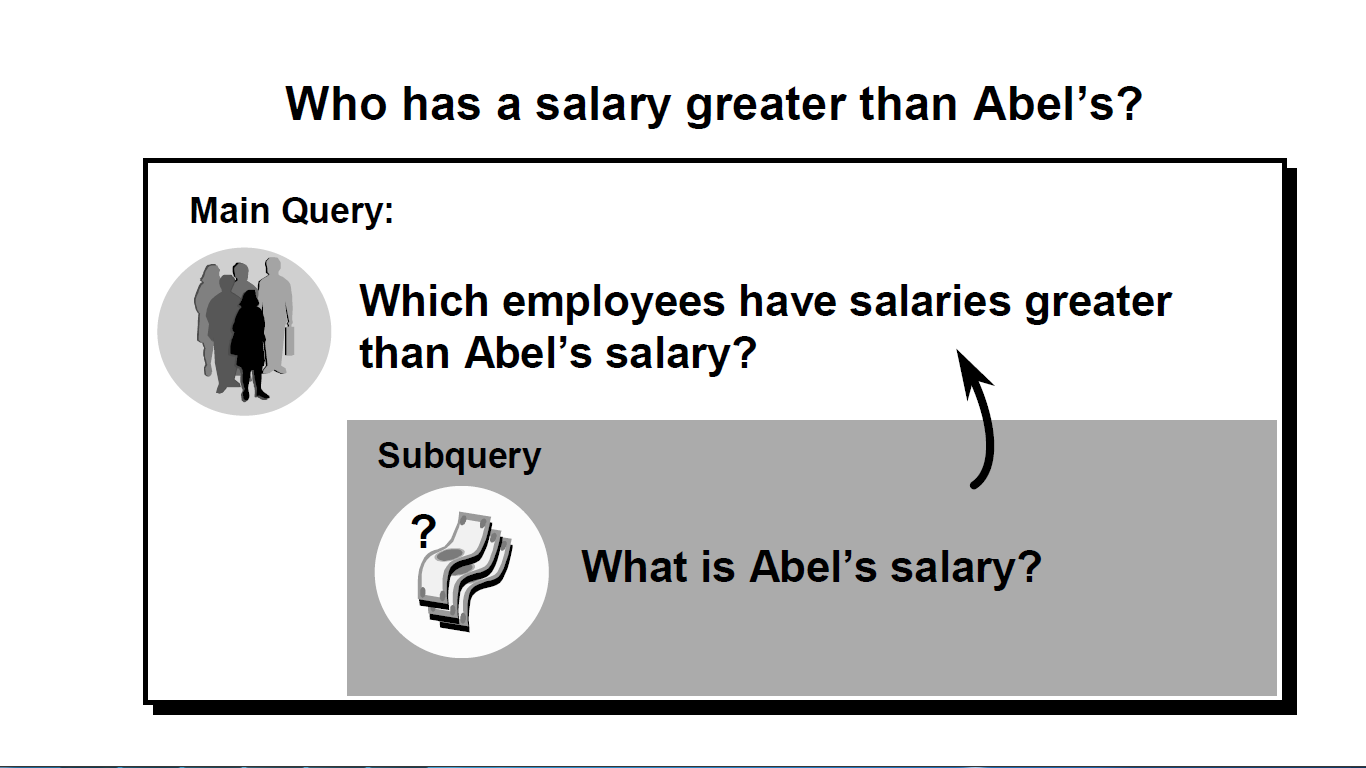
|  |
| --- |
| **C:\Users\faisal\Pictures\NSU_pic_download\n91267046457_2661.jpg**  **CSE 311L(Database Management System)**  **LAB-Week 05 (Part A)** |

**Subqueries**

Topics:

* Using a Subquery to Solve a Problem
* Subquery Syntax
* Single-Row Subqueries
* Executing Single-Row Subqueries
* Using Group Functions in a Subquery

**Who has a salary greater than Abel’s?**

****

**Using a Subquery**

SELECT last\_name

FROM emps

WHERE salary >(SELECT salary

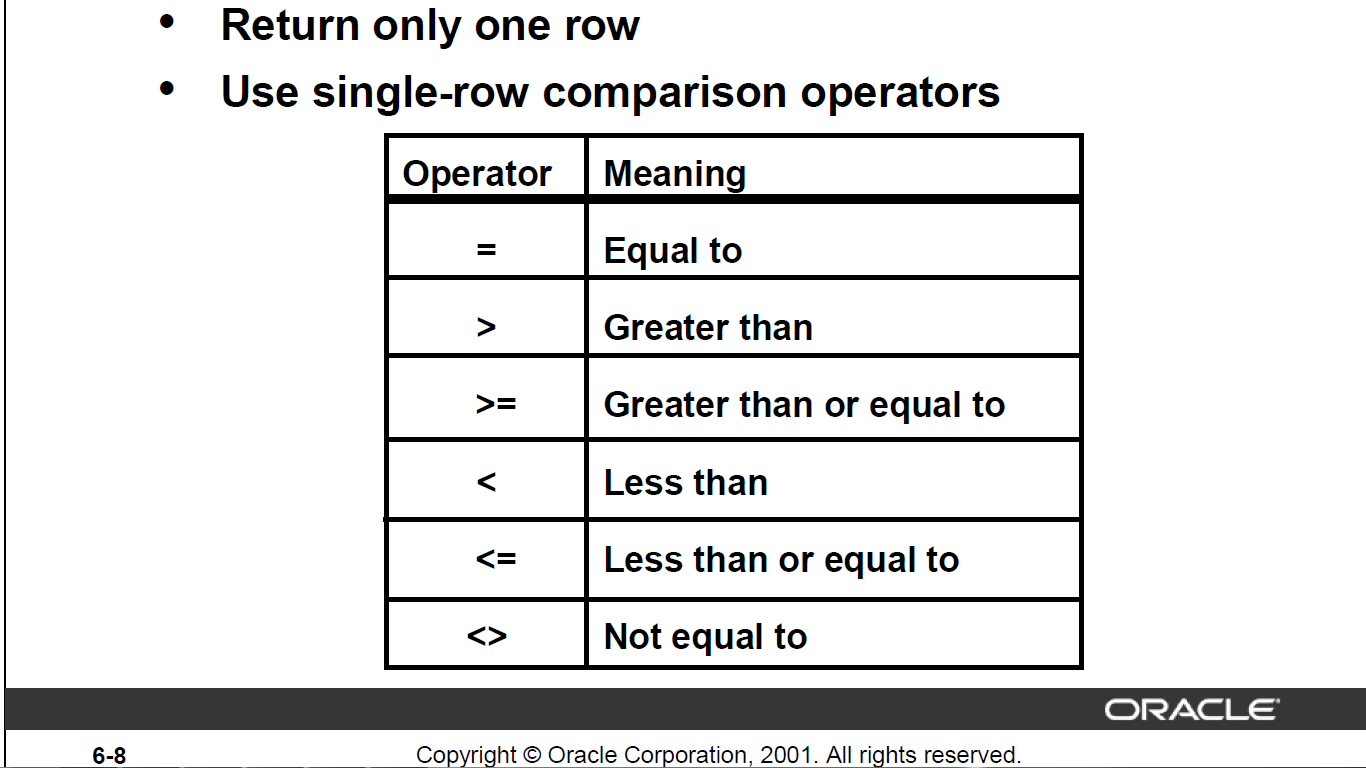
FROM emps

WHERE last\_name = ’Abel’);



**Single-Row Subqueries**

* Return only one row
* Use single-row comparison operators



**Executing Single-Row Subqueries**

SELECT last\_name, job\_id, salary

FROM emps

WHERE job\_id =

(SELECT job\_id

FROM emps

WHERE employee\_id = 141)

AND salary >

(SELECT salary

FROM emps

WHERE employee\_id = 143);

****

**Using Group Functions in a Subquery**

SELECT last\_name, job\_id, salary

FROM emps

WHERE salary =

(SELECT MIN(salary)

FROM emps);



**Activity 01:**

Write a query to display the last name and hire date of any employee in the same

department as Zlotkey. Exclude Zlotkey.



**Activity 02:**

Create a query to display the employee numbers and last names of all employees who earn more

than the average salary. Sort the results in ascending order of salary.

.



**Subqueries**

Topics:

* Single-row operator with multiple-row subquery
* Multiple-Row Subqueries
* Using the ANY Operator
* Using the ALL Operator

**What is Wrong with this Statement?**

SELECT employee\_id, last\_name

FROM emps

WHERE salary =

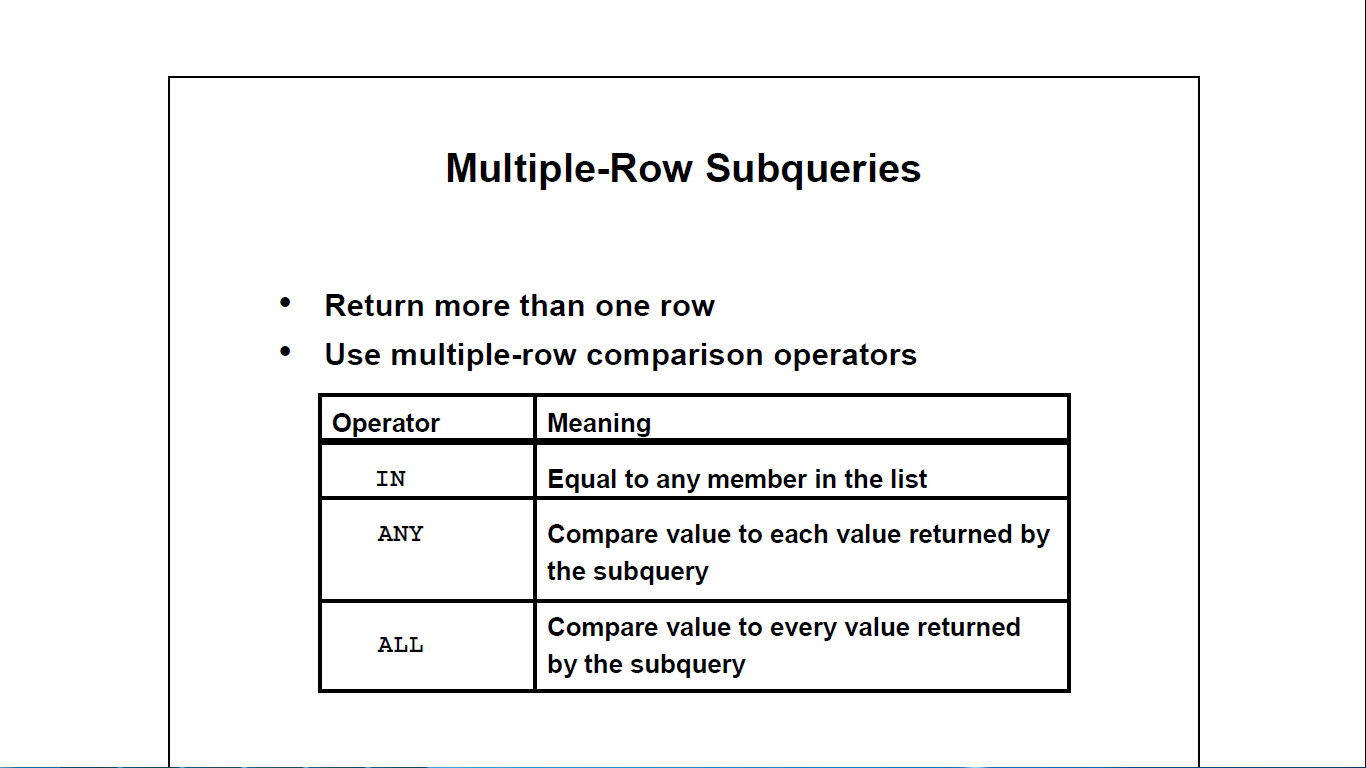
(SELECT MIN(salary)

FROM emps

GROUP BY department\_id);

**Multiple-Row Subqueries**

* Return more than one row
* Use multiple-row comparison operators



**Using the ANY Operator**

SELECT employee\_id, last\_name, job\_id, salary

FROM emps

WHERE salary < ANY

(SELECT salary

FROM emps

WHERE job\_id = ’IT\_PROG’)

AND job\_id <> ’IT\_PROG’;



**Using the ALL Operator**

SELECT employee\_id, last\_name, job\_id, salary

FROM emps

WHERE salary < ALL

(SELECT salary

FROM emps

WHERE job\_id = ’IT\_PROG’)

AND job\_id <> ’IT\_PROG’;



**Activity 01:**

Display the last name and salary of every employee who reports to King.



**Activity 02:**

Write a query to display the employee numbers, last names, and salaries of all employees who earn more than the average salary and who work in a department with any employee with a *u* in their name.



|  |
| --- |
| **C:\Users\faisal\Pictures\NSU_pic_download\n91267046457_2661.jpg**  **CSE 311L(Database Management System)**  **LAB-Week 05 (Part B)**  **))** |

**Manipulating Data**

Topics:

* Copying Rows from Another Table
* Updating Rows in a Table
* Updating Rows Based on Another Table
* Example of Merging Rows

**Copying Rows from Another Table**

INSERT INTO sales\_reps(id, name, salary, commission\_pct)

SELECT employee\_id, last\_name, salary, commission\_pct

FROM emps

WHERE job\_id LIKE ’%REP%’;

**Updating Rows in a Table**

UPDATE emps

SET department\_id = 70

WHERE employee\_id = 113;

**Updating Rows Based on Another Table**

UPDATE copy\_emp

SET department\_id =(SELECT department\_id

FROM emps

WHERE employee\_id = 100)

WHERE job\_id = (SELECT job\_id

FROM emps

WHERE employee\_id = 200);

**Example of Merging Rows**

MERGE INTO copy\_emp c

USING emps e

ON (c.employee\_id = e.employee\_id)

WHEN MATCHED THEN

UPDATE SET

c.first\_name = e.first\_name,

c.last\_name = e.last\_name,

c.email = e.email,

c.phone\_number = e.phone\_number,

c.hire\_date = e.hire\_date,

c.job\_id = e.job\_id,

c.salary = e.salary,

c.commission\_pct = e.commission\_pct,

c.manager\_id = e.manager\_id,

c.department\_id = e.department\_id

WHEN NOT MATCHED THEN

INSERT VALUES(e.employee\_id, e.first\_name, e.last\_name,

e.email, e.phone\_number, e.hire\_date, e.job\_id,

e.salary, e.commission\_pct, e.manager\_id,

e.department\_id);

**Activity 01:**

Create a table that has some fields similar to employees table. Then insert 5 rows to the new table. Afterwards, merge the new table and the employees table to a new table based on employee number.

|  |
| --- |
| **C:\Users\faisal\Pictures\NSU_pic_download\n91267046457_2661.jpg**  **CSE 311L(Database Management System)**  **LAB-Week 05 (Part C)** |

**Managing Tables**

Topics:

* The ALTER TABLE Statement
* Adding a Column
* Modifying a Column
* Dropping a Column
* Changing the Name of an Object
* Truncating a Table
* Add PRIMARY KEY/ FOREIGN KEY constraints
* CREATE VIEW

**The ALTER TABLE Statement**

Use the ALTER TABLE statement to:

* Add a new column
* Modify an existing column
* Define a default value for the new column
* Drop a column

**Adding a Column**

ALTER TABLE dept80

ADD (job\_id VARCHAR2(9));

**Modifying a Column**

ALTER TABLE dept80

MODIFY (last\_name VARCHAR2(30));

**Dropping a Column**

ALTER TABLE dept80

DROP COLUMN job\_id;

**Dropping a Table**

DROP TABLE dept80;

**Changing the Name of an Object**

RENAME dept TO detail\_dept;

**Truncating a Table**

TRUNCATE TABLE detail\_dept**;**

**Add PRIMARY KEY/ FOREIGN KEY constraints**

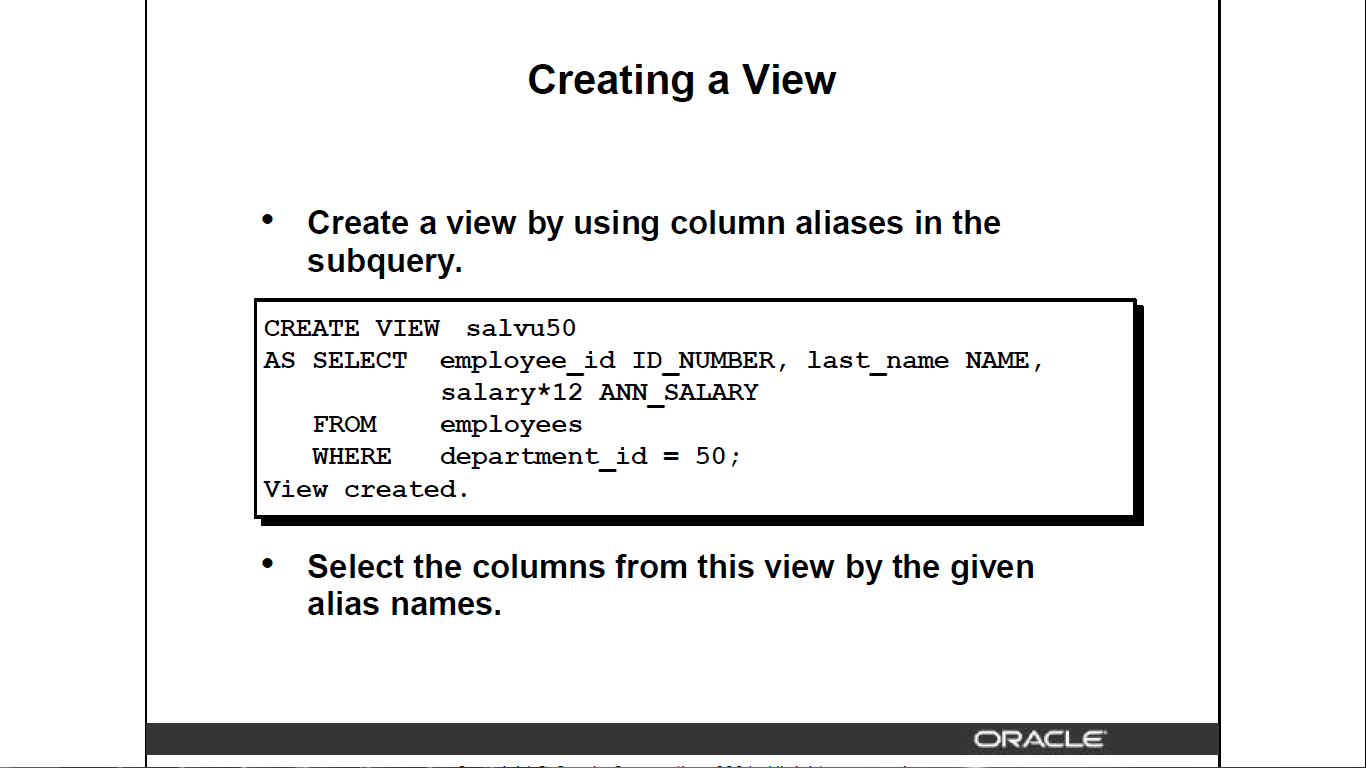
ALTER TABLE emps

ADD CONSTRAINT emp\_manager\_fk

FOREIGN KEY(manager\_id)

REFERENCES emps(employee\_id);

**Creating a View**



**Activity 01:**

Create the EMP table based on the following table instance chart..



a. Modify the EMP table to allow for longer employee last names. Confirm your modification.

b. Create the EMPLOYEES2 table based on the structure of the EMPLOYEES table. Include only the EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, SALARY, and DEPARTMENT\_ID

columns. Name the columns in your new table ID, FIRST\_NAME, LAST\_NAME, SALARY ,

and DEPT\_ID, respectively.

c. Drop the EMP table.

d. Rename the EMPLOYEES2 table as EMP.

e. Drop the FIRST\_NAME column from the EMP table. Confirm your modification by checking

the description of the table.