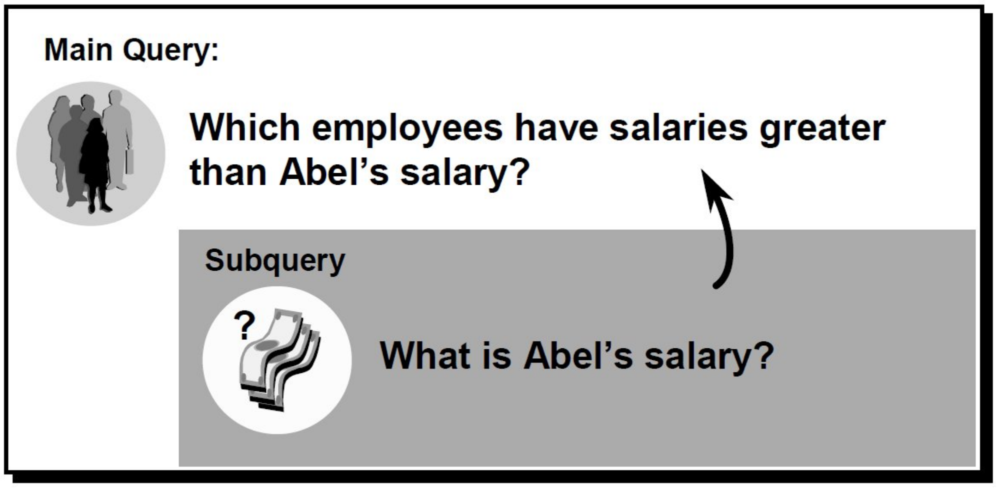
CSE 311L(Database Management System)

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');
```

LAST_NAME
King
Kochhar
De Haan
Hartstein
Higgins

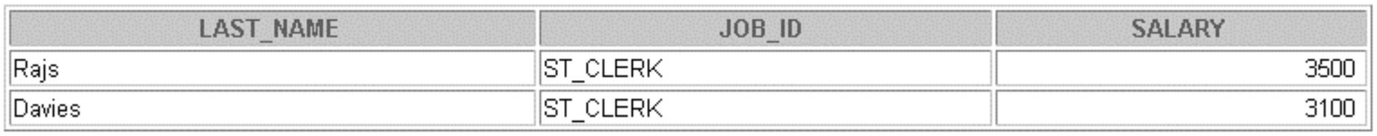
Single-Row Subqueries

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

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to

Executing Single-Row Subqueries

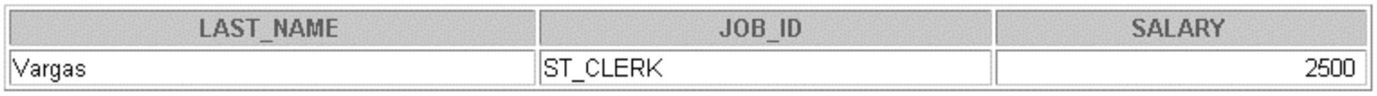
```
SELECT last_name, job_id, salary
FROM emps
WHERE job_id =
          (SELECT job_id
           FROM emps
           WHERE employee_id = 141)
AND salary >
          (SELECT salary
           FROM emps
           WHERE employee_id = 143);
```



LAST_NAME	JOB_ID	SALARY
Rajs	ST_CLERK	3500
Davies	ST_CLERK	3100

Using Group Functions in a Subquery

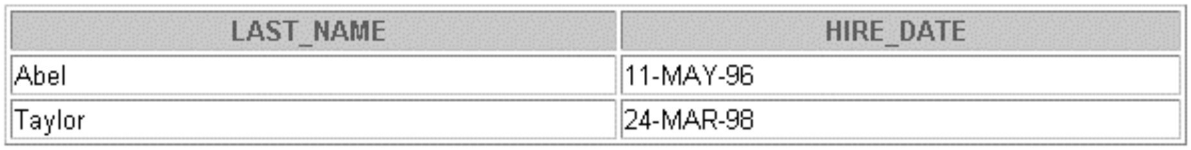
```
SELECT last_name, job_id, salary
FROM emps
WHERE salary =
               (SELECT MIN(salary)
                FROM emps);
```



LAST_NAME	JOB_ID	SALARY
Vargas	ST_CLERK	2500

Activity 01:

Write a query to display the last name and hire date of any employee in the same department as Zlotkey. Exclude Zlotkey.



LAST_NAME	HIRE_DATE
Abel	11-MAY-96
Taylor	24-MAR-98

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.

.

EMPLOYEE_ID	LAST_NAME	SALARY
103	Hunold	9000
149	Zlotkey	10500
174	Abel	11000
205	Higgins	12000
201	Hartstein	13000
101	Kochhar	17000
102	De Haan	17000
100	King	24000

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

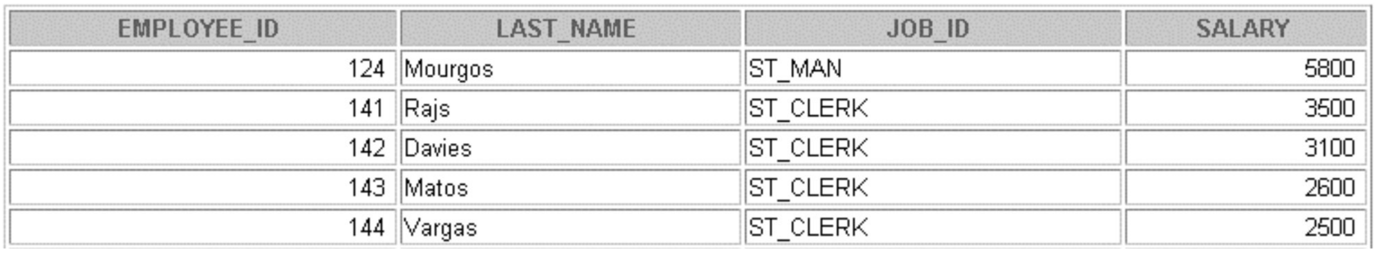
- Return more than one row
- Use multiple-row comparison operators

Operator	Meaning
IN	Equal to any member in the list
ANY	Compare value to each value returned by the subquery
ALL	Compare value to every value returned by the subquery

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')
```

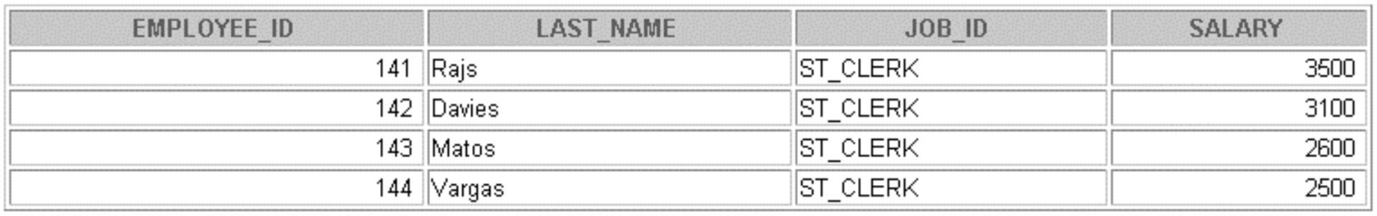
```
AND job_id <> 'IT_PROG';
```



EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
124	Mourgos	ST_MAN	5800
141	Rajs	ST_CLERK	3500
142	Davies	ST_CLERK	3100
143	Matos	ST_CLERK	2600
144	Vargas	ST_CLERK	2500

Using the ALL Operator

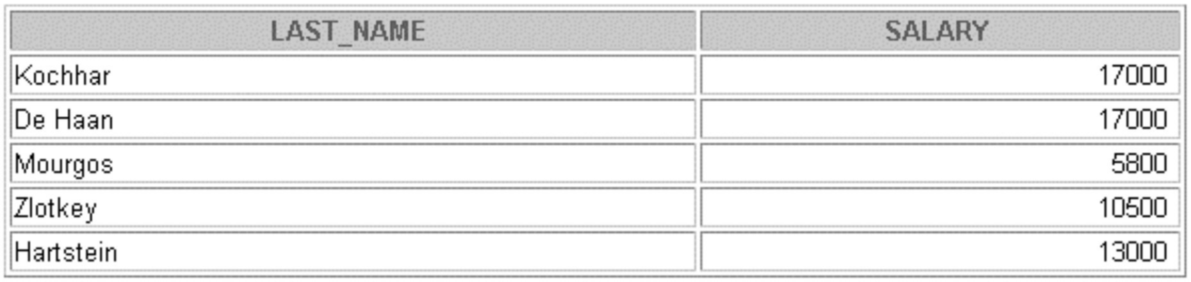
```
SELECT employee_id, last_name, job_id, salary
FROM emps
WHERE salary < ALL
                    (SELECT salary
                     FROM emps
                     WHERE job_id = 'IT_PROG')
AND job_id <> 'IT_PROG';
```



EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
141	Rajs	ST_CLERK	3500
142	Davies	ST_CLERK	3100
143	Matos	ST_CLERK	2600
144	Vargas	ST_CLERK	2500

Activity 01:

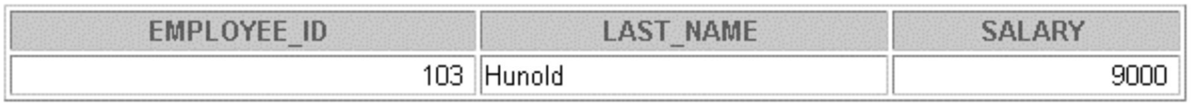
Display the last name and salary of every employee who reports to King.



LAST_NAME	SALARY
Kochhar	17000
De Haan	17000
Mourgos	5800
Zlotkey	10500
Hartstein	13000

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.



EMPLOYEE_ID	LAST_NAME	SALARY
103	Hunold	9000

CSE 311L(Database Management System)

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.

CSE 311L(Database Management System)

Managing Tables

Topics:

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);
```

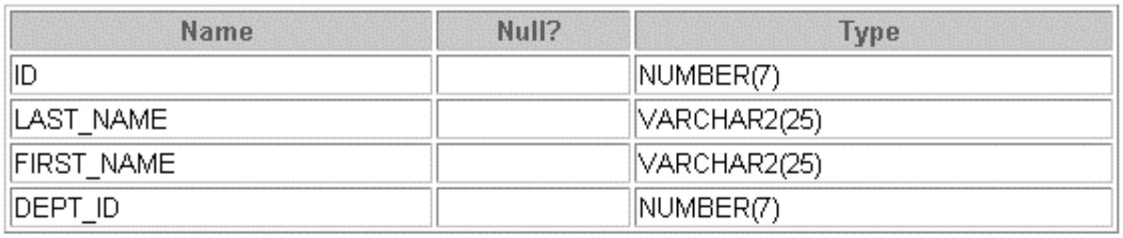
Creating a View

- **Create a view by using column aliases in the subquery.**

```
CREATE VIEW  salvu50
AS SELECT  employee_id ID_NUMBER, last_name NAME,
           salary*12 ANN_SALARY
   FROM    employees
   WHERE   department_id = 50;
View created.
```

Activity 01:

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



Name	Null?	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)

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.