Payroll Management System (PMS)

Payroll Management DBMS

S. C., K. G., E. N.

Group 26

Toronto Metropolitan University

Database Systems I

CPS510-092

27th November 2024

Table of Contents

Phase I

A1 - Application Description

1 Introduction

With modern businesses demanding a fast-paced work environment, managing payrolls with the highest accuracy and efficiency has become an important part in employee satisfaction. The service Payroll Management System (PMS) aims to efficiently process and manage payroll. PMS allows payroll managers to securely organize relevant information in order to create a comprehensive statement for employees. The utilization of PMS assists in automating payroll processes while maintaining accuracy in compliance with relevant tax and labour laws.

2 System Design

- What type of applications would each user need? To which user category would each belong and what type of interface would they need?
    - Employees: These users can only request to view information from their own employee profiles. Applications include:
        - Requesting information about profile related to their the employee id associated with their account
    - Managers: These users have permissions to view and update employee hours, as well as query data about certain employees as well, such as employees that are currently being managed by them. Applications include:
        - View and update time tables
        - Query employee data
    - Human Resources: These users can enter data that reflect the data associated with each employee, and update any relevant changes to other fields. Applications can include:
        - Creating a profile for a new employee
        - Get specific information about an employee and make changes to desired fields
        - Update changes to other tables as needed
        - Enter the student grades for a section

- Identify some informal queries and update operations that you would expect to apply to the database
    - (Query) List all: employees/employee data
    - (Query) What is the hourly rate of employee X/ number of hours worked by employee X
    - (Update) Insert Employee with Data = … (e.g. name)

  - ○ (Update) Total hours worked in current/previous week by employee X

- What types of system functions will be included in this system?

  Employees

  - ➢ View Employment Information (Employee Profile)
    - ○ (Query) Type, will use Employee ID to access the employee information within the “Employee” table assuming each employee ID is unique.
  - ➢ View Shift Information (Payroll)
    - ○ (Query) Type, access and display the most recent Payroll using the employee number.

  Managers

  - ➢ View Employee Under Management Information (Employee Profile)
    - ○ (Query) Type, will display the full list of employees under the management of the current manager (Requires Job Distinct ID).
  - ➢ View Employee Shift Information (Payroll).
    - ○ (Query) Type, will display the full payroll information of an employee under the current manager.
  - ➢ Update Employee Shift Information
    - ○ (Update) Type, allows the manager to update the current employee’s payroll information.
  - ➢ Terminate Employee
    - ○ (Update) Type, the manager can change the employee’s Employment Status to “Terminated” and subsequently “Hourly Rate” defaults to 0.00.

  Human Resources(HR)

  - ➢ View All Employees
    - ○ (Query) Type, will display the full list of employees under the company (Requires Job Distinct ID).
  - ➢ Create New Employee Profile
    - ○ (Update) Type, will create a new instance of an employee and job.
  - ➢ Update Employee Information
    - ○ (Update) Type, will update current instances of employees and jobs or allow for the population of a new instance.

Employee

Employee Number	Name		Date of Birth			SIN	Address	Employment Status	Hourly Rate ($)
	Last	First	YYYY	MM	DD				
1	Smith	John	2012	09	13	12345	123 Main Road. A1B 2C3	Active	16.55
2	Doe	Bob	2009	12	31	23456	234 Side Road. D4E 5F6	Terminated	0.00

Job

Job ID	Job Name	Description	Base Pay	Salary Range
1	Manager	Manages team	16.55	10000
2	Secretary	Office Management	17.20	11000

Payroll

Employee Number	Applicable Date	Pay Period		Paycode	Number of Hours	Multiplier
		Start	End			
1	2024-10-04	2024-09-13	2024-09-27	Regular	80	1.0
2	2024-10-04	2024-09-13	2024-09-27	Stat	65	2.0

Deductions

Employe	Pay Period	Deductible Type	Deductible amount

e Number	Start	End		($)
1	2024-09-27	2024-09-13	Tax	4789
2	2024-09-27	2024-09-13	Insurance	5987

Document

Employee Number	Document Type	Document Name	Issue Date	Expiry Date
1	Employment Contract	Contract_2024ver	2024-09-13	2024-09-14
2	Identification	Driver's License	2024-09-12	2024-09-13

2.1 Entity Relationships

- Specify the relationships among the records of the database
  - Each EMPLOYEE is related to a JOB record.
  - Each EMPLOYEE is related to a DOCUMENT record.
  - Each EMPLOYEE is related to a DEDUCTIONS record.

2.2 Integrity Constraints

- Cite some examples of integrity constraints that you think can apply to the database
  - The 'Employee Number' should be a unique value for each EMPLOYEE record (key constraint).
  - The 'SIN' should be a unique value for each EMPLOYEE record (key constraint).
  - The 'Job ID' should be a unique value for each JOB record (key constraint).

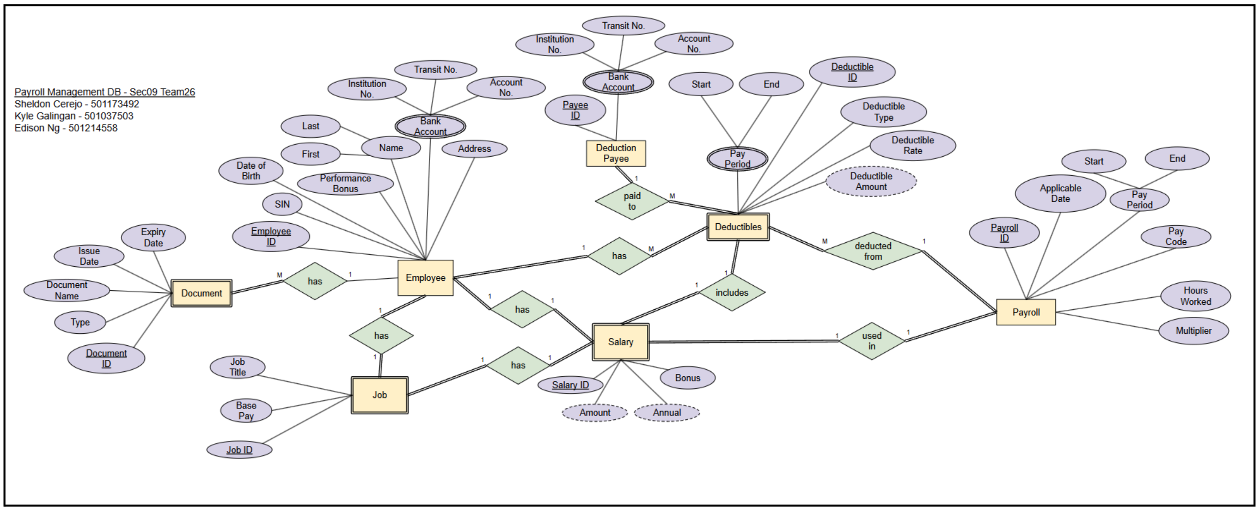
- ○ The value for 'Employee Number' must also exist in DEDUCTION and DOCUMENT records (domain constraint).

Conclusion

The Payroll Management System is an ideal solution for companies in hectic work environments that will improve managing payrolls as well as documenting employee and employer data. The PMS will store, read and retrieve large amounts of relevant company data and automate the payroll process through the use of queries. By automating the payment process, companies are provided with a convenient way to manage payrolls effectively and reliably.

A1 was a rough outline and brainstorming of the project we planned on making. Many of the core ideas such as some of the main entity tables were thought of here. Several user types were also taken into consideration when creating the DBMS to see how different users would be able to interact with different levels of the DBMS. Many of the rough entity relationships were also brainstormed here whereas more detailed relationships including attributes would be drawn up within A2 where the ER diagram was made.

A2 - Entity Relationship Diagram



Above is the implementation of the DBMS Entity-Relationship(ER) Diagram. The diagram consists of seven total entities with strong entities EMPLOYEE, DEDUCTION PAYEE and PAYROLL and SALARY, JOB, DOCUMENT and DEDUCTIBLES being weak entities. Strong entities are independent from other entities, whereas weak entities are dependent on at least one other entity.

Each entity consists of several attributes, key attributes are indicated by the underlined attributes within the purple ovals. These attributes uniquely identify the entity from the rest of the entities in the DBMS.

Double-circled attributes symbolize multivariable attributes, these are indicative of attributes that an entity can possess several instances of. For example, Bank Account under the EMPLOYEE entity, any employee can have several bank accounts.

The most important part of the ER diagram is the actual relationship between entities. The relationships are as follows, each EMPLOYEE has many documents, has one job, one salary, and many deductibles. Every SALARY is associated with one job, and includes one deductible and is used in one payroll. Many DEDUCTIBLES are deducted from one payroll, one deductible is included in one salary, many deductibles are paid to one deduction payee.

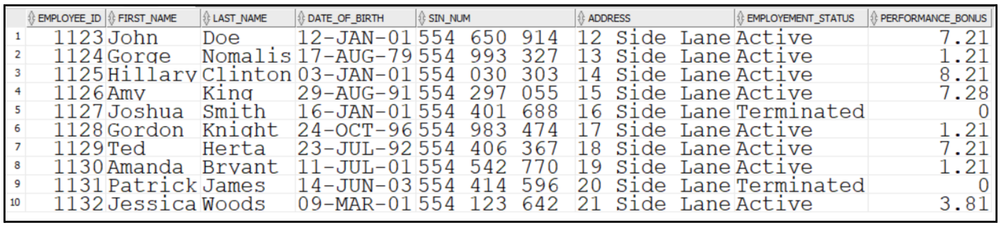
A2 was a deepdive into what the final attributes and tables were. It was in this assignment where we determined how we would want to create our tables and the types of entities and relationships they would have with one another. This diagram became a blueprint for the creation of the entire DBMS and all future assignments stem from this diagram.

A3 - Schema Design

(Source Code of A3 in Appendix)

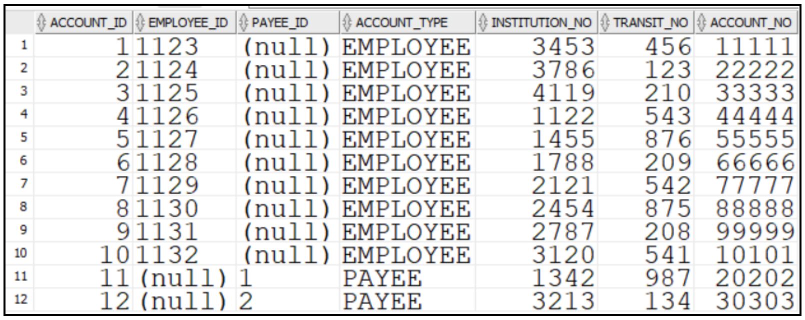
Created Tables

EMPLOYEE



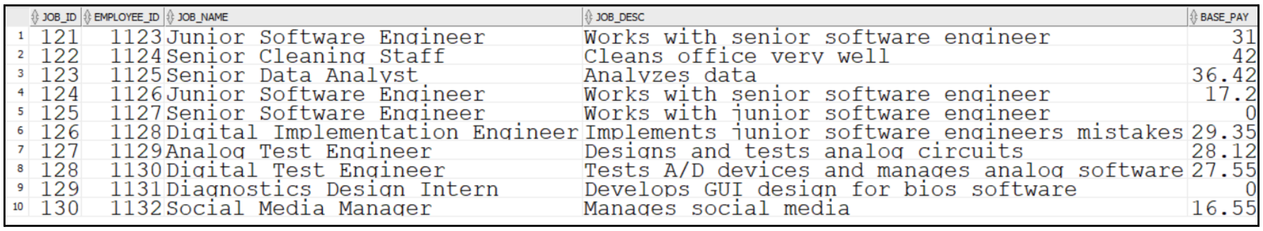
	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DATE_OF_BIRTH	SIN_NUM	ADDRESS	EMPLOYEMENT_STATUS	PERFORMANCE_BONUS
1	1123	John	Doe	12-JAN-01	554 650 914	12 Side Lane	Active	7.21
2	1124	Gorge	Nomalis	17-AUG-79	554 993 327	13 Side Lane	Active	1.21
3	1125	Hillary	Clinton	03-JAN-01	554 030 303	14 Side Lane	Active	8.21
4	1126	Amy	King	29-AUG-91	554 297 055	15 Side Lane	Active	7.28
5	1127	Joshua	Smith	16-JAN-01	554 401 688	16 Side Lane	Terminated	0
6	1128	Gordon	Knight	24-OCT-96	554 983 474	17 Side Lane	Active	1.21
7	1129	Ted	Herta	23-JUL-92	554 406 367	18 Side Lane	Active	7.21
8	1130	Amanda	Bryant	11-JUL-01	554 542 770	19 Side Lane	Active	1.21
9	1131	Patrick	James	14-JUN-03	554 414 596	20 Side Lane	Terminated	0
10	1132	Jessica	Woods	09-MAR-01	554 123 642	21 Side Lane	Active	3.81

BANK_ACCOUNT



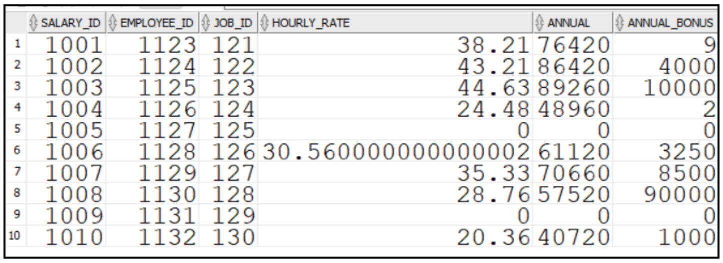
	ACCOUNT_ID	EMPLOYEE_ID	PAYEE_ID	ACCOUNT_TYPE	INSTITUTION_NO	TRANSIT_NO	ACCOUNT_NO
1	1	1123	(null)	EMPLOYEE	3453	456	11111
2	2	1124	(null)	EMPLOYEE	3786	123	22222
3	3	1125	(null)	EMPLOYEE	4119	210	33333
4	4	1126	(null)	EMPLOYEE	1122	543	44444
5	5	1127	(null)	EMPLOYEE	1455	876	55555
6	6	1128	(null)	EMPLOYEE	1788	209	66666
7	7	1129	(null)	EMPLOYEE	2121	542	77777
8	8	1130	(null)	EMPLOYEE	2454	875	88888
9	9	1131	(null)	EMPLOYEE	2787	208	99999
10	10	1132	(null)	EMPLOYEE	3120	541	10101
11	11	(null)	1	PAYEE	1342	987	20202
12	12	(null)	2	PAYEE	3213	134	30303

JOB_TABLE



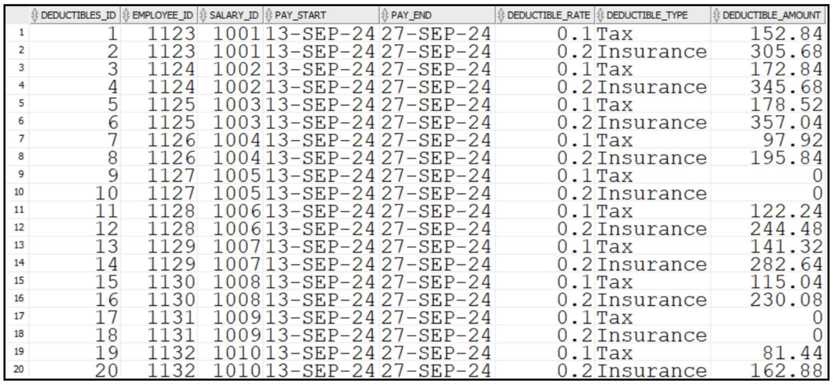
	JOB_ID	EMPLOYEE_ID	JOB_NAME	JOB_DESC	BASE_PAY
1	121	1123	Junior Software Engineer	Works with senior software engineer	31
2	122	1124	Senior Cleaning Staff	Cleans office very well	42
3	123	1125	Senior Data Analyst	Analyzes data	36.42
4	124	1126	Junior Software Engineer	Works with senior software engineer	17.2
5	125	1127	Senior Software Engineer	Works with junior software engineer	0
6	126	1128	Digital Implementation Engineer	Implements junior software engineers mistakes	29.35
7	127	1129	Analog Test Engineer	Designs and tests analog circuits	28.12
8	128	1130	Digital Test Engineer	Tests A/D devices and manages analog software	27.55
9	129	1131	Diagnostics Design Intern	Develops GUI design for bios software	0
10	130	1132	Social Media Manager	Manages social media	16.55

SALARY



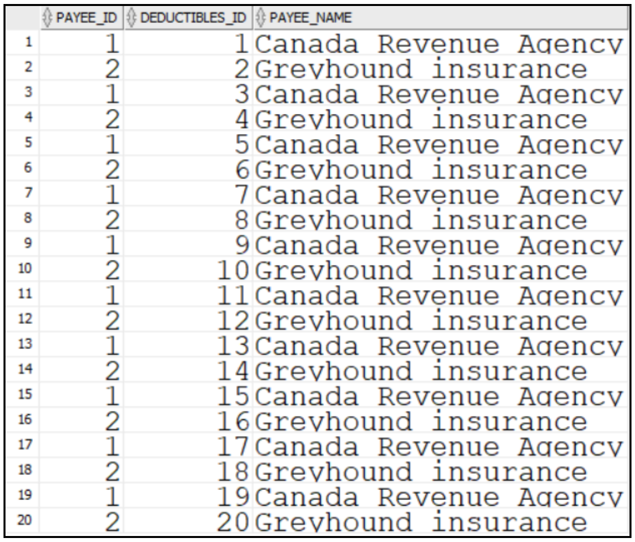
	SALARY_ID	EMPLOYEE_ID	JOB_ID	HOURLY_RATE	ANNUAL	ANNUAL_BONUS
1	1001	1123	121	38.21	76420	9
2	1002	1124	122	43.21	86420	4000
3	1003	1125	123	44.63	89260	10000
4	1004	1126	124	24.48	48960	2
5	1005	1127	125	0	0	0
6	1006	1128	126	30.560000000000002	61120	3250
7	1007	1129	127	35.33	70660	8500
8	1008	1130	128	28.76	57520	90000
9	1009	1131	129	0	0	0
10	1010	1132	130	20.36	40720	1000

DEDUCTIBLES



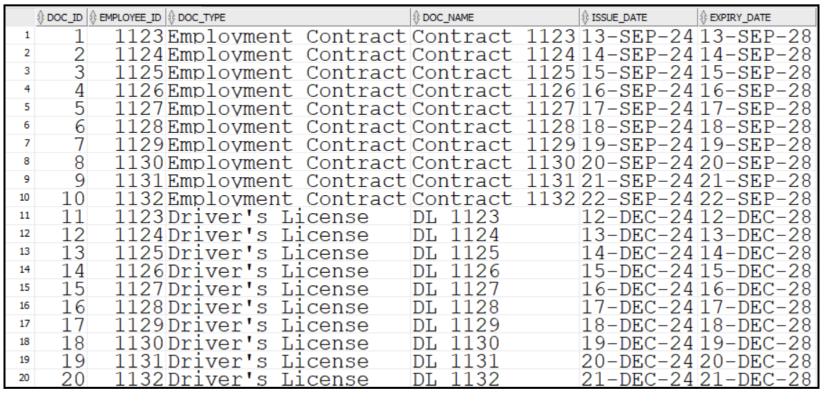
	DEDUCTIBLES_ID	EMPLOYEE_ID	SALARY_ID	PAY_START	PAY_END	DEDUCTIBLE_RATE	DEDUCTIBLE_TYPE	DEDUCTIBLE_AMOUNT
1	1	1123	1001	13-SEP-24	27-SEP-24	0.1	Tax	152.84
2	2	1123	1001	13-SEP-24	27-SEP-24	0.2	Insurance	305.68
3	3	1124	1002	13-SEP-24	27-SEP-24	0.1	Tax	172.84
4	4	1124	1002	13-SEP-24	27-SEP-24	0.2	Insurance	345.68
5	5	1125	1003	13-SEP-24	27-SEP-24	0.1	Tax	178.52
6	6	1125	1003	13-SEP-24	27-SEP-24	0.2	Insurance	357.04
7	7	1126	1004	13-SEP-24	27-SEP-24	0.1	Tax	97.92
8	8	1126	1004	13-SEP-24	27-SEP-24	0.2	Insurance	195.84
9	9	1127	1005	13-SEP-24	27-SEP-24	0.1	Tax	0
10	10	1127	1005	13-SEP-24	27-SEP-24	0.2	Insurance	0
11	11	1128	1006	13-SEP-24	27-SEP-24	0.1	Tax	122.24
12	12	1128	1006	13-SEP-24	27-SEP-24	0.2	Insurance	244.48
13	13	1129	1007	13-SEP-24	27-SEP-24	0.1	Tax	141.32
14	14	1129	1007	13-SEP-24	27-SEP-24	0.2	Insurance	282.64
15	15	1130	1008	13-SEP-24	27-SEP-24	0.1	Tax	115.04
16	16	1130	1008	13-SEP-24	27-SEP-24	0.2	Insurance	230.08
17	17	1131	1009	13-SEP-24	27-SEP-24	0.1	Tax	0
18	18	1131	1009	13-SEP-24	27-SEP-24	0.2	Insurance	0
19	19	1132	1010	13-SEP-24	27-SEP-24	0.1	Tax	81.44
20	20	1132	1010	13-SEP-24	27-SEP-24	0.2	Insurance	162.88

DEDUCTIBLES_PAYEE



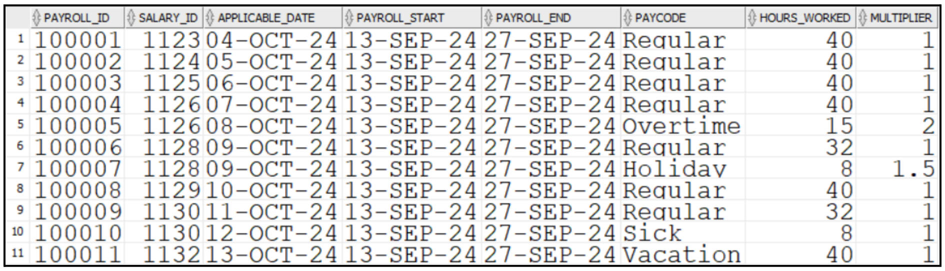
	PAYEE_ID	DEDUCTIBLES_ID	PAYEE_NAME
1	1	1	Canada Revenue Agency
2	2	2	Greyhound insurance
3	1	3	Canada Revenue Agency
4	2	4	Greyhound insurance
5	1	5	Canada Revenue Agency
6	2	6	Greyhound insurance
7	1	7	Canada Revenue Agency
8	2	8	Greyhound insurance
9	1	9	Canada Revenue Agency
10	2	10	Greyhound insurance
11	1	11	Canada Revenue Agency
12	2	12	Greyhound insurance
13	1	13	Canada Revenue Agency
14	2	14	Greyhound insurance
15	1	15	Canada Revenue Agency
16	2	16	Greyhound insurance
17	1	17	Canada Revenue Agency
18	2	18	Greyhound insurance
19	1	19	Canada Revenue Agency
20	2	20	Greyhound insurance

DOCUMENT_TABLE



	DOC_ID	EMPLOYEE_ID	DOC_TYPE	DOC_NAME	ISSUE_DATE	EXPIRY_DATE
1	1	1123	Employment Contract	Contract 1123	13-SEP-24	13-SEP-28
2	2	1124	Employment Contract	Contract 1124	14-SEP-24	14-SEP-28
3	3	1125	Employment Contract	Contract 1125	15-SEP-24	15-SEP-28
4	4	1126	Employment Contract	Contract 1126	16-SEP-24	16-SEP-28
5	5	1127	Employment Contract	Contract 1127	17-SEP-24	17-SEP-28
6	6	1128	Employment Contract	Contract 1128	18-SEP-24	18-SEP-28
7	7	1129	Employment Contract	Contract 1129	19-SEP-24	19-SEP-28
8	8	1130	Employment Contract	Contract 1130	20-SEP-24	20-SEP-28
9	9	1131	Employment Contract	Contract 1131	21-SEP-24	21-SEP-28
10	10	1132	Employment Contract	Contract 1132	22-SEP-24	22-SEP-28
11	11	1123	Driver's License	DL 1123	12-DEC-24	12-DEC-28
12	12	1124	Driver's License	DL 1124	13-DEC-24	13-DEC-28
13	13	1125	Driver's License	DL 1125	14-DEC-24	14-DEC-28
14	14	1126	Driver's License	DL 1126	15-DEC-24	15-DEC-28
15	15	1127	Driver's License	DL 1127	16-DEC-24	16-DEC-28
16	16	1128	Driver's License	DL 1128	17-DEC-24	17-DEC-28
17	17	1129	Driver's License	DL 1129	18-DEC-24	18-DEC-28
18	18	1130	Driver's License	DL 1130	19-DEC-24	19-DEC-28
19	19	1131	Driver's License	DL 1131	20-DEC-24	20-DEC-28
20	20	1132	Driver's License	DL 1132	21-DEC-24	21-DEC-28

PAYROLL



	PAYROLL_ID	SALARY_ID	APPLICABLE_DATE	PAYROLL_START	PAYROLL_END	PAYCODE	HOURS_WORKED	MULTIPLIER
1	100001	1123	04-OCT-24	13-SEP-24	27-SEP-24	Regular	40	1
2	100002	1124	05-OCT-24	13-SEP-24	27-SEP-24	Regular	40	1
3	100003	1125	06-OCT-24	13-SEP-24	27-SEP-24	Regular	40	1
4	100004	1126	07-OCT-24	13-SEP-24	27-SEP-24	Regular	40	1
5	100005	1126	08-OCT-24	13-SEP-24	27-SEP-24	Overtime	15	2
6	100006	1128	09-OCT-24	13-SEP-24	27-SEP-24	Regular	32	1
7	100007	1128	09-OCT-24	13-SEP-24	27-SEP-24	Holiday	8	1.5
8	100008	1129	10-OCT-24	13-SEP-24	27-SEP-24	Regular	40	1
9	100009	1130	11-OCT-24	13-SEP-24	27-SEP-24	Regular	32	1
10	100010	1130	12-OCT-24	13-SEP-24	27-SEP-24	Sick	8	1
11	100011	1132	13-OCT-24	13-SEP-24	27-SEP-24	Vacation	40	1

Retrieving Employee Details

	SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME, DATE_OF_BIRTH, SIN_NUM,

Query	ADDRESS, EMPLOYMENT_STATUS, PERFORMANCE_BONUS
	FROM EMPLOYEE
	WHERE EMPLOYEE_ID = 1;

Relational Algebra	πEMPLOYEE_ID,FIRST_NAME,LAST_NAME,DATE_OF_BIRTH,SIN_NUM,ADDRESS,EMPLOYMENT_STATUS,PERFORMANCE_BONUS(σEMPLOYEE_ID=1(EMPLOYEE))

Retrieving Bank Account Details

Query	SELECT ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID, ACCOUNT_TYPE, INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO
	FROM BANK_ACCOUNT
	WHERE ACCOUNT_ID = 1 AND ACCOUNT_NO = 11111;

Relational Algebra	πACCOUNT_ID,EMPLOYEE_ID,PAYEE_ID,ACCOUNT_TYPE,INSTITUTION_NO,TRANSIT_NO,ACCOUNT_NO(σACCOUNT_ID=1∧ACCOUNT_NO=11111(BANK_ACCOUNT))

Retrieving Job Information

Query	SELECT JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC, BASE_PAY
	FROM JOB_TABLE
	WHERE EMPLOYEE_ID = 1 AND JOB_ID = 1;

Relational Algebra	πJOB_ID,EMPLOYEE_ID,JOB_NAME,JOB_DESC,BASE_PAY(σEMPLOYEE_ID=1∧JOB_ID=1(JOB_TABLE))

Retrieving Salary Details

Query	SELECT SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE, ANNUAL, ANNUAL_BONUS
	FROM SALARY
	WHERE HOURLY_RATE = 10000 AND SALARY_ID = 1;

Relational Algebra	πSALARY_ID,EMPLOYEE_ID,JOB_ID,HOURLY_RATE,ANNUAL,ANNUAL_BONUS(σHOURLY_RATE=10000∧SALARY_ID=1(SALAR

	Y))

Retrieving Deductibles

Query	SELECT DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID, PAY_START, PAY_END, DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE, DEDUCTIBLE_AMOUNT
	FROM DEDUCTIBLES
	WHERE DEDUCTIBLES_ID = 1;

Relational Algebra	πDEDUCTIBLES_ID,EMPLOYEE_ID,SALARY_ID,PAY_START,PAY_END,DEDUCTIBLE_RATE,DEDUCTIBLE_TYPE,DEDUCTIBLE_AMOUNT(σDEDUCTIBLES_ID=1(DEDUCTIBLES))

Retrieving Deductibles Payee

Query	SELECT PAYEE_ID, DEDUCTIBLES_ID, PAYEE_NAME
	FROM DEDUCTIBLES_PAYEE
	WHERE PAYEE_ID = 101 AND DEDUCTIBLES_ID = 1;

Relational Algebra	πPAYEE_ID,DEDUCTIBLES_ID,PAYEE_NAME(σPAYEE_ID=101∧DEDUCTIBLES_ID=1 (DEDUCTIBLES_PAYEE))

Retrieving Document Details

Query	SELECT DOC_ID, EMPLOYEE_ID, DOC_TYPE, DOC_NAME, ISSUE_DATE, EXPIRY_DATE
	FROM DOCUMENT_TABLE
	WHERE EMPLOYEE_ID = 1;

Relational Algebra	πDOC_ID,EMPLOYEE_ID,DOC_TYPE,DOC_NAME,ISSUE_DATE,EXPIRY_DATE(σEMPLOYEE_ID=1(DOCUMENT_TABLE))

Retrieving Payroll Details

Query	SELECT PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID, APPLICABLE_DATE, PAYROLL_START, PAYROLL_END, PAYCODE, HOURS_WORKED, MULTIPLIER

	FROM PAYROLL
	WHERE PAYROLL_ID = 1 AND HOURS_WORKED > 0;

Relational Algebra	πPAYROLL_ID,SALARY_ID,DEDUCTIBLES_ID,APPLICABLE_DATE,PAYROLL_START,PAYROLL_END,PAYCODE,HOURS_WORKED,MULTIPLIER(σPAYROLL_ID=1∧HOURS_WORKED>0(PAYROLL))

In A3 we took the ER/EER Diagram of A2 and derived them into tables using oracle. We created the desired results of the table creation on Microsoft Excel and then using that as a baseline replicated it by using Oracle's SQL Developer. On top of the table creation insert queries were used to populate the tables as well as drop queries to drop all tables if needed. 8 Additional SELECT queries were used to retrieve and display several pieces of information. These are the queries displayed above, accompanying them are their respective relational algebra equivalents. An example explanation can be found below:

There are two main parts of the RA equivalent, the Projection Pi(π) and Selection sigma(σ). The Projection is the selected columns in the table and the Selection is the condition that will be required to retrieve the desired data.
πEMPLOYEE_ID,FIRST_NAME,LAST_NAME,DATE_OF_BIRTH,SIN_NUM,ADDRESS,EMPLOYMENT_STATUS,PERFORMANCE_BONUS(σEMPLOYEE_ID=1(EMPLOYEE))
In this case the RA can be read as the Projection of EMPLOYEE_ID,FIRST_NAME,LAST_NAME,DATE_OF_BIRTH,SIN_NUM,ADDRESS,EMPLOYMENT_STATUS,PERFORMANCE_BONUS With a selection of EMPLOYEE_ID=1(EMPLOYEE) This simply selects the data from the columns EMPLOYEE_ID,FIRST_NAME,LAST_NAME,DATE_OF_BIRTH,SIN_NUM,ADDRESS,EMPLOYMENT_STATUS,PERFORMANCE_BONUS For the employee with an EMPLOYEE_ID=1 from the EMPLOYEE table.

Phase II

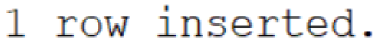
A4 - Demo of Designing Views/Simple Queries

Update “Bill, Clinton” to “Hillary, Clinton”

Query	UPDATE EMPLOYEE
	SET FIRST_NAME = ‘Hillary’ WHERE FIRST_NAME = ‘Bill’ AND LAST_NAME = ‘Clinton’ AND EMPLOYEE_ID = 1125;

Relational Algebra	(EMPLOYEE−σFIRST_NAME=′Bill′∧LAST_NAME=′Clinton′∧EMPLOYEE_ID=1125(EMPLOYEE))∪{(1125,′Hillary′,′Clinton′,...)}
Output	1 row updated.

Insert Employee “Mac, Book”

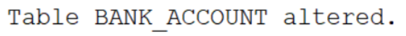
Query	INSERT INTO EMPLOYEE (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, DATE_OF_BIRTH, SIN_NUM, ADDRESS, EMPLOYEMENT_STATUS, PERFORMANCE_BONUS) VALUES (1420, 'Mac', 'Book', TO_DATE('2006-5-16','YYYY-MM-DD'), '553 993 327', '10 Down Lane', 'Active', 0.00);
Relational Algebra	EMPLOYEE∪{(1420,′Mac′,′Book′,DATE(′2006−05−16′),′553993327′,′10DownLane′,′Active′,0.00)}
Output	 1 row inserted.

Delete Employee “Mac, Book”

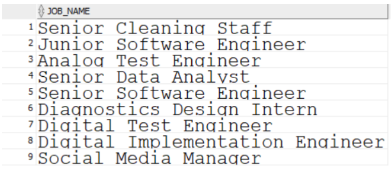
Query	DELETE FROM EMPLOYEE WHERE EMPLOYEE_ID = 1420 AND FIRST_NAME = 'Mac' AND LAST_NAME = 'Book';
Relational Algebra	EMPLOYEE−σEMPLOYEE_ID=1420∧FIRST_NAME=′Mac′∧LAST_NAME=′Book′(EMPLOYEE)
Output	1 row deleted.

Alter BANK_ACCOUNT

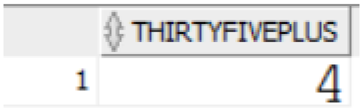
Query	ALTER TABLE BANK_ACCOUNT ADD BANK_NAME VARCHAR(20);

Relational Algebra	BANK_ACCOUNT′=πACCOUNT_ID,EMPLOYEE_ID,PAYEE_ID,ACCOUNT_TYPE,INSTITUTION_NO,TRANSIT_NO,ACCOUNT_NO,BANK_NAME(BANK_ACCOUNT)
Output	 Table BANK_ACCOUNT altered.

SELECT DISTINCT JOB_NAME

Query	SELECT DISTINCT JOB_NAME from JOB_TABLE;
Relational Algebra	πJOB_NAME(JOB_TABLE)
Output	 JOB_NAME 1 Senior Cleaning Staff 2 Junior Software Engineer 3 Analog Test Engineer 4 Senior Data Analyst 5 Senior Software Engineer 6 Diagnostics Design Intern 7 Digital Test Engineer 8 Digital Implementation Engineer 9 Social Media Manager

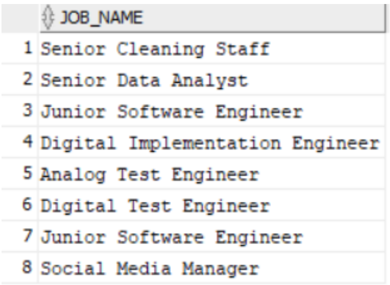
SELECT HOURLY_RATE > 35

Query	SELECT COUNT(*) AS ThirtyFivePlus FROM SALARY WHERE HOURLY_RATE > 35.00;
Relational Algebra	∣σHOURLY_RATE>35.00(SALARY)∣
Output	 THIRTYFIVEPLUS 1 4

SELECT job_name and ORDER BY

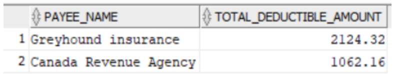
Query	SELECT
	    job_name
	FROM
	    job_table
	WHERE

	base_pay > 0
	ORDER BY
	base_pay DESC;

Relational Algebra	πjob_name(σbase_pay>0(JOB_TABLE))
Output	 JOB_NAME 1 Senior Cleaning Staff 2 Senior Data Analyst 3 Junior Software Engineer 4 Digital Implementation Engineer 5 Analog Test Engineer 6 Digital Test Engineer 7 Junior Software Engineer 8 Social Media Manager

SELECT, JOIN, GROUP BY

Query	SELECT
	dp.payee_name,
	SUM(d.deductible_amount) AS total_deductible_amount
	FROM
	deductibles_payee dp
	JOIN deductibles d ON dp.deductibles_id = d.deductibles_id
	GROUP BY
	dp.payee_name;

Relational Algebra	γpayee_name,SUM(deductible_amount)(πpayee_name,deductible_amount(DED_UCTIBLES_PAYEE⋈deductibles_idDED_UCTIBLES))
Output	 PAYEE_NAME TOTAL_DEDUCTIBLE_AMOUNT 1 Greyhound insurance 2124.32 2 Canada Revenue Agency 1062.16

View overthirty

	CREATE VIEW overthirty AS SELECT e.employee_id, e.first_name, e.last_name,

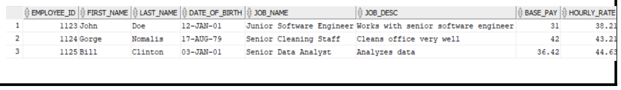
Query

```
    e.date_of_birth,
    j.job_name,
    j.job_desc,
    j.base_pay,
    s.hourly_rate
FROM
         employee e
   JOIN job_table j ON e.employee_id = j.employee_id
    JOIN salary    s ON j.employee_id = s.employee_id
WHERE
    j.base_pay > 30;
```

Relational Algebra

πemployee_id,first_name,last_name,date_of_birth,job_name,job_desc,base_pay,hourly_rate(σbase_pay>30((EMPLOYEE⋉employee_idJOB_TABLE)⋉employee_idSALARY))

Output

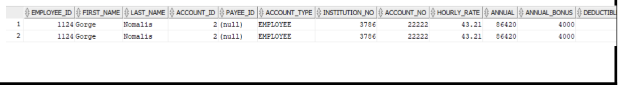


	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DATE_OF_BIRTH	JOB_NAME	JOB_DESC	BASE_PAY	HOURLY_RATE
1	1123	John	Doe	12-JAN-01	Junior Software Engineer	Works with senior software engineer	31	38.21
2	1124	Gorge	Nomalis	17-AUG-79	Senior Cleaning Staff	Cleans office very well	42	43.21
3	1125	Bill	Clinton	03-JAN-01	Senior Data Analyst	Analyzes data	36.42	44.63

View employeesalary

Query

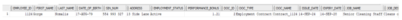
```
CREATE VIEW employeesalary AS
    SELECT
        emp.employee_id,
        emp.first_name,
        emp.last_name,
        b.account_id,
        b.payee_id,
        b.account_type,
        b.institution_no,
        b.account_no,
        s.hourly_rate,
        s.annual,
        s.annual_bonus,
        d.deductible_rate,
        d.deductible_type,
        d.deductible_amount,
        d.pay_start,
        d.pay_end
    FROM
        employee     emp
        RIGHT JOIN bank_account b ON
emp.employee_id = b.employee_id
        JOIN salary       s ON emp.employee_id
= s.employee_id
        JOIN deductibles  d ON
emp.employee_id = d.employee_id
                              AND s.salary_id =
d.salary_id
    WHERE
```

	emp.employee_id = 1124;
Relational Algebra	πemployee_id,first_name,last_name,account_id,payee_id,account_type,institution_no,account_no,hourly_rate,annual,annual_bonus,deductible_rate,deductible_type,deductible_amount,pay_start,pay_end(σemp.employee_id=1124(SALARY_DEDUCTIBLES_JOIN))
Output	

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	ACCOUNT_ID	PAYEE_ID	ACCOUNT_TYPE	INSTITUTION_NO	ACCOUNT_NO	HOURLY_RATE	ANNUAL	ANNUAL_BONUS	DEDUCTIBL
1	1124	Gorge	Nomalis	2	(null)	EMPLOYEE	3786	22222	43.21	86420	4000	
2	1124	Gorge	Nomalis	2	(null)	EMPLOYEE	3786	22222	43.21	86420	4000	

View employeecontract

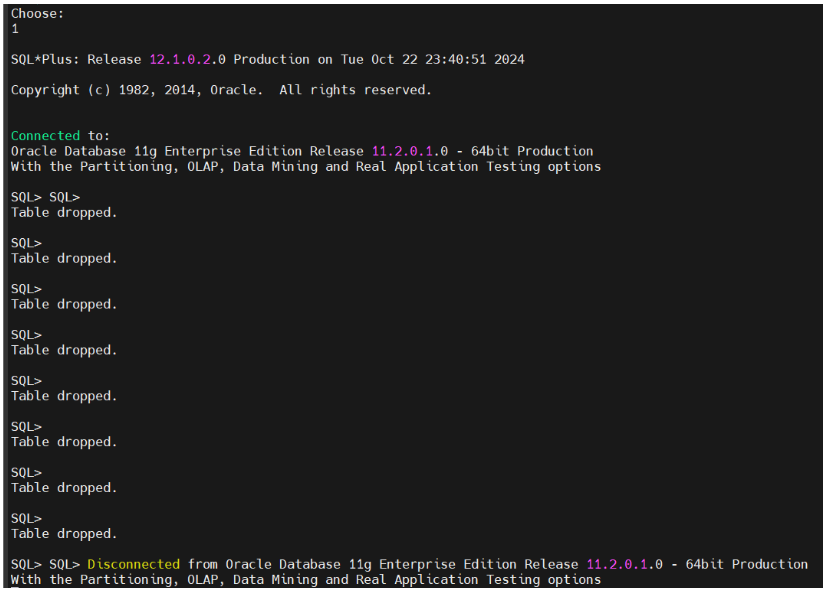
Query	CREATE VIEW employeecontract AS SELECT emp.employee_id, emp.first_name, emp.last_name, emp.date_of_birth, emp.sin_num, emp.address, emp.employment_status, emp.performance_bonus, doc.doc_id, doc.doc_type, doc.doc_name, doc.issue_date, doc.expiry_date, jobs.job_name, jobs.job_desc, jobs.base_pay FROM employee emp JOIN document_table doc ON emp.employee_id = doc.employee_id JOIN job_table jobs ON jobs.employee_id = doc.employee_id WHERE emp.employee_id = 1124 AND doc.doc_type = 'Employment Contract';
Relational Algebra	πemployee_id,first_name,last_name,date_of_birth,sin_num,address,employment_status,performance_bonus,doc_id,doc_type,doc_name,iss

<table>
<tr><td></td><td>ue_date,expiry_date,job_name,job_desc,base_pay(σemp.employee_id=1124∧doc.doc_type='EmploymentContract'((EMPLOYEE⋈employee_idDOCUMENT_TABLE)⋈employee_idJOB_TABLE))</td></tr>
<tr><td>Output</td><td>
<pre>
   EMPLOYEE_ID FIRST_NAME LAST_NAME DATE_OF_BIRTH SIN_NUM   ADDRESS   EMPLOYMENT_STATUS PERFORMANCE_BONUS DOC_ID DOC_TYPE   DOC_NAME   ISSUE_DATE EXPIRY_DATE JOB_NAME   JOB_DES
1   1124 Gorge   Nomalis   17-AUG-79   554 993 327 13 Side Lane Active   1.21   2 Employment Contract Contract_1124 14-SEP-24 14-SEP-28   Senior Cleaning Staff Cleans c
</pre></td></tr>
</table>

In A4 we started coming up with more specialized queries to isolate specific types of information. SELECT was one of the main tools used to do so, SELECT and JOIN were used to merge and pick out specific data from multiple tables. This then paired with VIEWS allowed us to create a pseudo table that held that isolated data. This allowed us to improve the functionality of the app, users could now isolate a specific employee or view all the employees employment contracts using this feature.

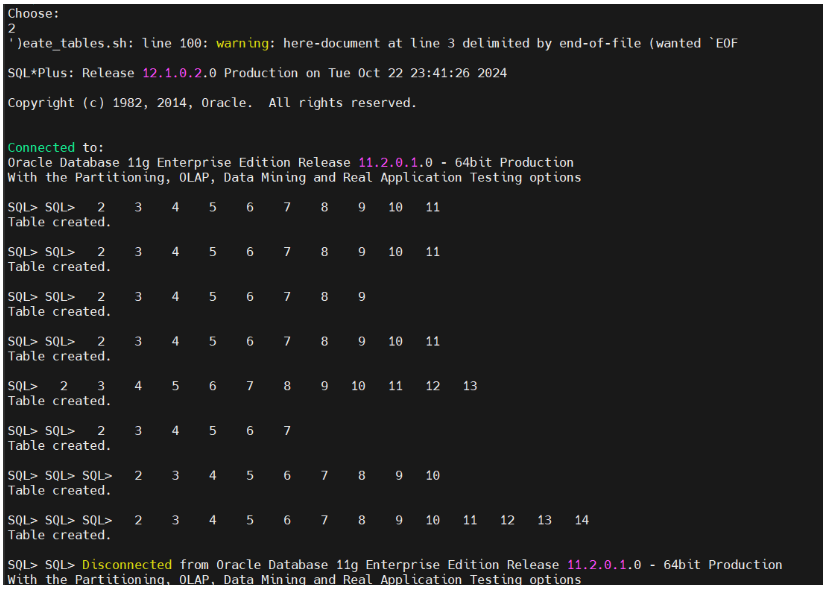
A5 - Demonstration of Advanced Queries by Unix Shell Implementation

(Bash File Code in Appendix)
Drop Tables



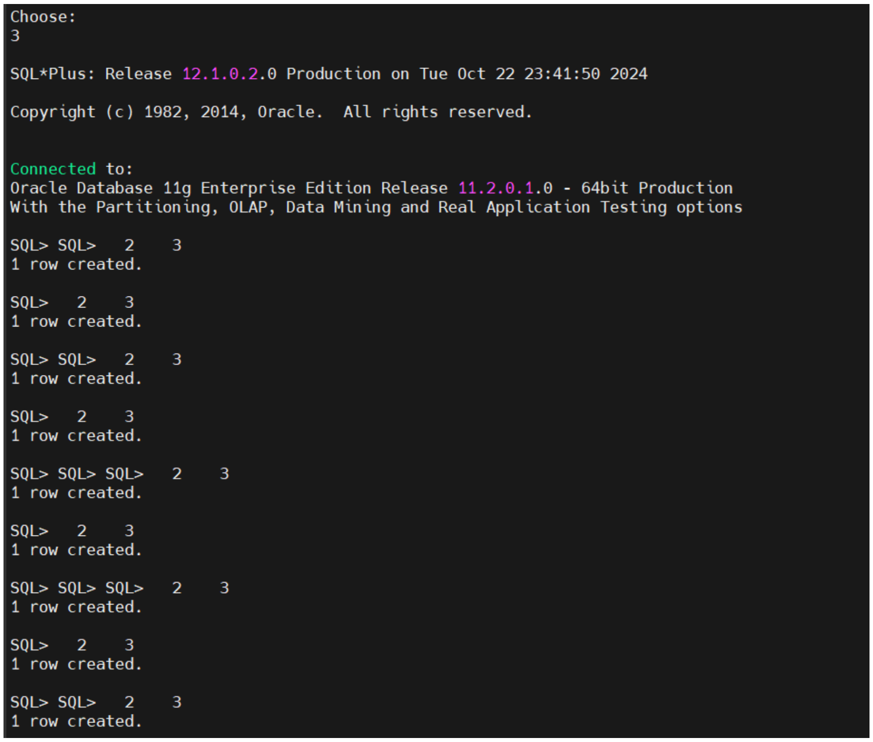
```
Choose:
1

SQL*Plus: Release 12.1.0.2.0 Production on Tue Oct 22 23:40:51 2024

Copyright (c) 1982, 2014, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> SQL>
Table dropped.

SQL>
Table dropped.

SQL>
Table dropped.

SQL>
Table dropped.

SQL>
Table dropped.

SQL>
Table dropped.

SQL>
Table dropped.

SQL>
Table dropped.

SQL> SQL> Disconnected from Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

Create Tables



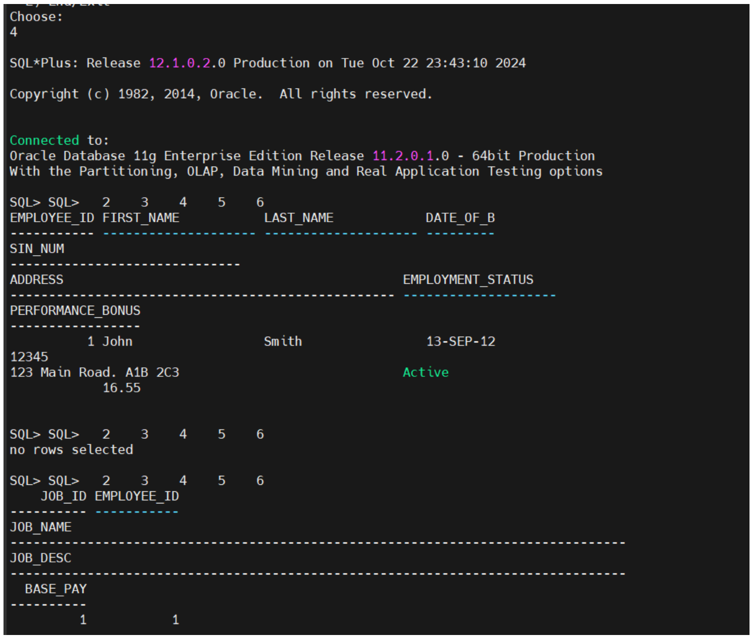
```
Choose:
2
')eate_tables.sh: line 100: warning: here-document at line 3 delimited by end-of-file (wanted `EOF

SQL*Plus: Release 12.1.0.2.0 Production on Tue Oct 22 23:41:26 2024

Copyright (c) 1982, 2014, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> SQL>   2    3    4    5    6    7    8    9   10   11
Table created.

SQL> SQL>   2    3    4    5    6    7    8    9   10   11
Table created.

SQL> SQL>   2    3    4    5    6    7    8    9
Table created.

SQL> SQL>   2    3    4    5    6    7    8    9   10   11
Table created.

SQL>   2    3    4    5    6    7    8    9   10   11   12   13
Table created.

SQL> SQL>   2    3    4    5    6    7
Table created.

SQL> SQL> SQL>   2    3    4    5    6    7    8    9   10
Table created.

SQL> SQL> SQL>   2    3    4    5    6    7    8    9   10   11   12   13   14
Table created.

SQL> SQL> Disconnected from Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

Populate Tables

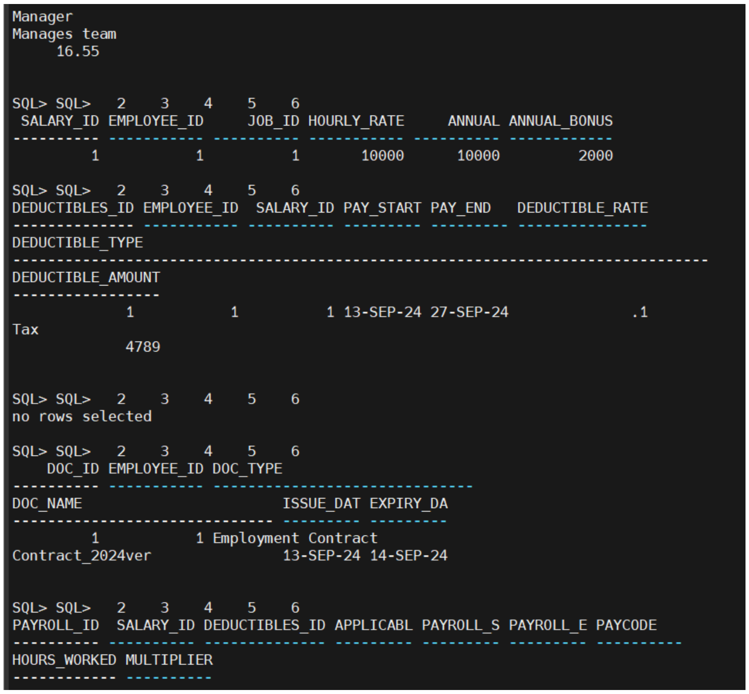


```
Choose:
3

SQL*Plus: Release 12.1.0.2.0 Production on Tue Oct 22 23:41:50 2024

Copyright (c) 1982, 2014, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> SQL>   2    3
1 row created.

SQL>   2    3
1 row created.

SQL> SQL>   2    3
1 row created.

SQL>   2    3
1 row created.

SQL> SQL> SQL>   2    3
1 row created.

SQL>   2    3
1 row created.

SQL> SQL> SQL>   2    3
1 row created.

SQL>   2    3
1 row created.

SQL> SQL>   2    3
1 row created.
```

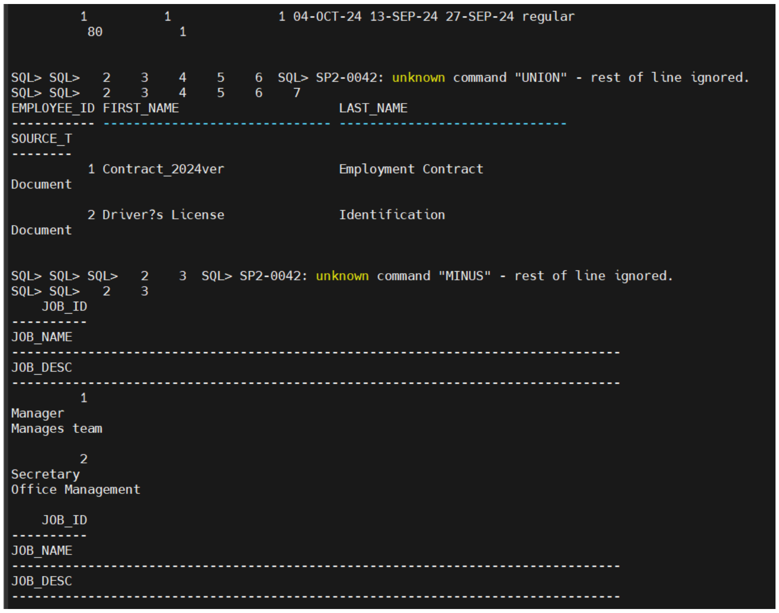
Queries



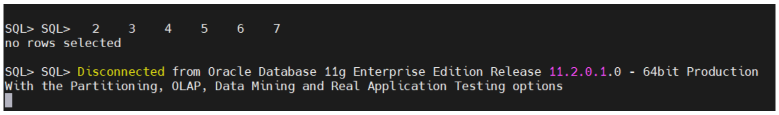
```
Choose:
4

SQL*Plus: Release 12.1.0.2.0 Production on Tue Oct 22 23:43:10 2024

Copyright (c) 1982, 2014, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> SQL>   2    3    4    5    6
EMPLOYEE_ID FIRST_NAME           LAST_NAME            DATE_OF_B
----------- -------------------- -------------------- ---------
SIN_NUM
------------------------------
ADDRESS                                            EMPLOYMENT_STATUS
-------------------------------------------------- --------------------
PERFORMANCE_BONUS
-----------------
          1 John                 Smith                13-SEP-12
12345
123 Main Road. A1B 2C3                             Active
            16.55


SQL> SQL>   2    3    4    5    6
no rows selected

SQL> SQL>   2    3    4    5    6
    JOB_ID EMPLOYEE_ID
---------- -----------
JOB_NAME
--------------------------------------------------------------------------------
JOB_DESC
--------------------------------------------------------------------------------
  BASE_PAY
----------
         1           1
```



```
Manager
Manages team
     16.55


SQL> SQL>   2    3    4    5    6
 SALARY_ID EMPLOYEE_ID     JOB_ID HOURLY_RATE     ANNUAL ANNUAL_BONUS
---------- ----------- ---------- ----------- ---------- ------------
         1           1          1       10000      10000         2000

SQL> SQL>   2    3    4    5    6
DEDUCTIBLES_ID EMPLOYEE_ID  SALARY_ID PAY_START PAY_END   DEDUCTIBLE_RATE
-------------- ----------- ---------- --------- --------- ---------------
DEDUCTIBLE_TYPE
--------------------------------------------------------------------------------
DEDUCTIBLE_AMOUNT
-----------------
             1           1          1 13-SEP-24 27-SEP-24              .1
Tax
             4789


SQL> SQL>   2    3    4    5    6
no rows selected

SQL> SQL>   2    3    4    5    6
    DOC_ID EMPLOYEE_ID DOC_TYPE
---------- ----------- ------------------------------
DOC_NAME                       ISSUE_DAT EXPIRY_DA
------------------------------ --------- ---------
         1           1 Employment Contract
Contract_2024ver               13-SEP-24 14-SEP-24


SQL> SQL>   2    3    4    5    6
PAYROLL_ID  SALARY_ID DEDUCTIBLES_ID APPLICABL PAYROLL_S PAYROLL_E PAYCODE
---------- ---------- -------------- --------- --------- --------- ----------
HOURS_WORKED MULTIPLIER
------------ ----------
```



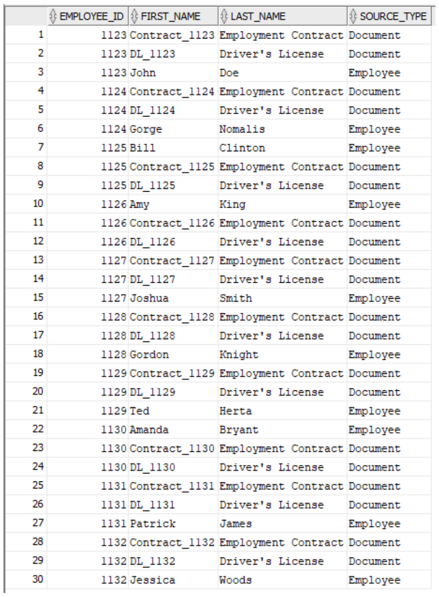
```
         1          1              1 04-OCT-24 13-SEP-24 27-SEP-24 regular
          80          1


SQL> SQL>   2    3    4    5    6  SQL> SP2-0042: unknown command "UNION" - rest of line ignored.
SQL> SQL>   2    3    4    5    6    7
EMPLOYEE_ID FIRST_NAME                     LAST_NAME
----------- ------------------------------ ------------------------------
SOURCE_T
--------
          1 Contract_2024ver               Employment Contract
Document

          2 Driver?s License               Identification
Document


SQL> SQL> SQL>   2    3  SQL> SP2-0042: unknown command "MINUS" - rest of line ignored.
SQL> SQL>   2    3
    JOB_ID
----------
JOB_NAME
--------------------------------------------------------------------------------
JOB_DESC
--------------------------------------------------------------------------------
         1
Manager
Manages team

         2
Secretary
Office Management

    JOB_ID
----------
JOB_NAME
--------------------------------------------------------------------------------
JOB_DESC
--------------------------------------------------------------------------------
```

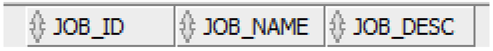


```
SQL> SQL>   2    3    4    5    6    7
no rows selected

SQL> SQL> Disconnected from Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

Advanced Queries

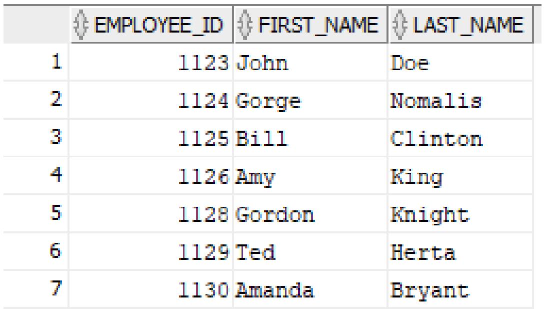
	CREATE VIEW employeecontract AS
	    SELECT
	        emp.employee_id,

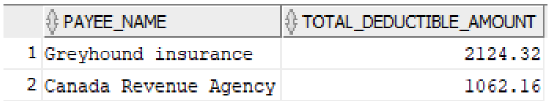
<table>
<tr>
<td>Query</td>
<td>emp.first_name,
emp.last_name,
emp.date_of_birth,
emp.sin_num,
emp.address,
emp.employment_status,
emp.performance_bonus,
doc.doc_id,
doc.doc_type,
doc.doc_name,
doc.issue_date,
doc.expiry_date,
jobs.job_name,
jobs.job_desc,
jobs.base_pay
FROM
employee emp
JOIN document_table doc ON emp.employee_id = doc.employee_id
JOIN job_table jobs ON jobs.employee_id = doc.employee_id
WHERE
emp.employee_id = 1124
AND doc.doc_type = 'Employment Contract';</td>
</tr>
<tr>
<td>Relational Algebra</td>
<td>(πEMPLOYEE_ID,FIRST_NAME,LAST_NAME,’Employee’ →SOURCE_TYPE(EMPLOYEE))∪(πEMPLOYEE_ID,DOC_NAME→FIRST_NAME,DOC_TYPE→LAST_NAME,’Document’ →SOURCE_TYPE(DOCUMENT_TABLE))</td>
</tr>
<tr>
<td>Output</td>
<td>
<table>
<tr><th></th><th>EMPLOYEE_ID</th><th>FIRST_NAME</th><th>LAST_NAME</th><th>SOURCE_TYPE</th></tr>
<tr><td>1</td><td>1123</td><td>Contract_1123</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>2</td><td>1123</td><td>DL_1123</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>3</td><td>1123</td><td>John</td><td>Doe</td><td>Employee</td></tr>
<tr><td>4</td><td>1124</td><td>Contract_1124</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>5</td><td>1124</td><td>DL_1124</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>6</td><td>1124</td><td>Gorge</td><td>Nomalis</td><td>Employee</td></tr>
<tr><td>7</td><td>1125</td><td>Bill</td><td>Clinton</td><td>Employee</td></tr>
<tr><td>8</td><td>1125</td><td>Contract_1125</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>9</td><td>1125</td><td>DL_1125</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>10</td><td>1126</td><td>Amy</td><td>King</td><td>Employee</td></tr>
<tr><td>11</td><td>1126</td><td>Contract_1126</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>12</td><td>1126</td><td>DL_1126</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>13</td><td>1127</td><td>Contract_1127</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>14</td><td>1127</td><td>DL_1127</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>15</td><td>1127</td><td>Joshua</td><td>Smith</td><td>Employee</td></tr>
<tr><td>16</td><td>1128</td><td>Contract_1128</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>17</td><td>1128</td><td>DL_1128</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>18</td><td>1128</td><td>Gordon</td><td>Knight</td><td>Employee</td></tr>
<tr><td>19</td><td>1129</td><td>Contract_1129</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>20</td><td>1129</td><td>DL_1129</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>21</td><td>1129</td><td>Ted</td><td>Herta</td><td>Employee</td></tr>
<tr><td>22</td><td>1130</td><td>Amanda</td><td>Bryant</td><td>Employee</td></tr>
<tr><td>23</td><td>1130</td><td>Contract_1130</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>24</td><td>1130</td><td>DL_1130</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>25</td><td>1131</td><td>Contract_1131</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>26</td><td>1131</td><td>DL_1131</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>27</td><td>1131</td><td>Patrick</td><td>James</td><td>Employee</td></tr>
<tr><td>28</td><td>1132</td><td>Contract_1132</td><td>Employment Contract</td><td>Document</td></tr>
<tr><td>29</td><td>1132</td><td>DL_1132</td><td>Driver's License</td><td>Document</td></tr>
<tr><td>30</td><td>1132</td><td>Jessica</td><td>Woods</td><td>Employee</td></tr>
</table>
</td>
</tr>
</table>

Query	(SELECT JOB_ID, JOB_NAME, JOB_DESC FROM JOB_TABLE) MINUS (SELECT JOB_ID, JOB_NAME, JOB_DESC FROM EMPLOYEE JOIN JOB_TABLE ON EMPLOYEE.EMPLOYEE_ID = JOB_TABLE.EMPLOYEE_ID);
Relational Algebra	(πJOB_ID,JOB_NAME,JOB_DESC(JOB_TABLE))−(πJOB_ID,JOB_NAME,JOB_DESC(EMPLOYEE⋈EMPLOYEE.EMPLOYEE_ID=JOB_TABLE.EMPLOYEE_IDJOB_TABLE))
Output	 JOB_ID JOB_NAME JOB_DESC

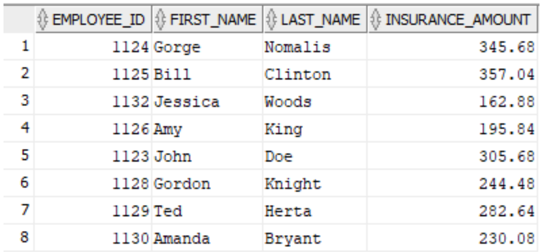
Query	SELECT EMPLOYEE_ID, FIRST_NAME,
	LAST_NAME
	FROM EMPLOYEE E
	WHERE EXISTS (
	SELECT 1
	FROM SALARY S
	WHERE S.EMPLOYEE_ID =
	E.EMPLOYEE_ID
	AND S.ANNUAL > 45000);

Relational Algebra	πEMPLOYEE_ID,FIRST_NAME,LAST_NAME(EMPLOYEE⋈EMPLOYEE.EMPLOYEE_ID=SALARY.EMPLOYEE_IDσSALARY.ANNUAL>45000(SALARY))

<table>
<tr><td>Output</td><td>
<table>
<tr><th></th><th>EMPLOYEE_ID</th><th>FIRST_NAME</th><th>LAST_NAME</th></tr>
<tr><td>1</td><td>1123</td><td>John</td><td>Doe</td></tr>
<tr><td>2</td><td>1124</td><td>Gorge</td><td>Nomalis</td></tr>
<tr><td>3</td><td>1125</td><td>Bill</td><td>Clinton</td></tr>
<tr><td>4</td><td>1126</td><td>Amy</td><td>King</td></tr>
<tr><td>5</td><td>1128</td><td>Gordon</td><td>Knight</td></tr>
<tr><td>6</td><td>1129</td><td>Ted</td><td>Herta</td></tr>
<tr><td>7</td><td>1130</td><td>Amanda</td><td>Bryant</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>Query</td><td><pre>
SELECT
   dp.payee_name,
   SUM(d.deductible_amount) AS
total_deductible_amount
FROM
       deductibles_payee dp
   JOIN deductibles d ON
dp.deductibles_id = d.deductibles_id
GROUP BY
   dp.payee_name;
</pre></td></tr>
<tr><td>Relational Algebra</td><td>γpayee_name;SUM(deductible_amount)→total_deductible_amount(deductibles_payee⋈deductibles_payee.deductibles_id=deductibles.deductibles_iddeductibles)</td></tr>
<tr><td>Output</td><td>
<table>
<tr><th></th><th>PAYEE_NAME</th><th>TOTAL_DEDUCTIBLE_AMOUNT</th></tr>
<tr><td>1</td><td>Greyhound insurance</td><td>2124.32</td></tr>
<tr><td>2</td><td>Canada Revenue Agency</td><td>1062.16</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>Query</td><td><pre>
SELECT
   e.employee_id,
   e.first_name,
   e.last_name,
   SUM(d.deductible_amount) AS
insurance_amount
FROM
       employee e
   JOIN deductibles d ON e.employee_id
= d.employee_id
WHERE
</pre></td></tr>
</table>

<table>
<tr><td></td><td>d.deductible_type = 'Insurance'
GROUP BY
e.employee_id,
e.first_name,
e.last_name
HAVING
SUM(d.deductible_amount) > 100;</td></tr>
<tr><td>Relational Algebra</td><td>σSUM(deductible_amount)>100(γemployee_id,first_name,last_name;SUM(deductible_amount)→insurance_amount(σdeductible_type=′Insurance′(employee⋈employee.employee_id=deductibles.employee_iddeductibles)))</td></tr>
<tr><td>Output</td><td>
<table>
<tr><th></th><th>EMPLOYEE_ID</th><th>FIRST_NAME</th><th>LAST_NAME</th><th>INSURANCE_AMOUNT</th></tr>
<tr><td>1</td><td>1124</td><td>Gorge</td><td>Nomalis</td><td>345.68</td></tr>
<tr><td>2</td><td>1125</td><td>Bill</td><td>Clinton</td><td>357.04</td></tr>
<tr><td>3</td><td>1132</td><td>Jessica</td><td>Woods</td><td>162.88</td></tr>
<tr><td>4</td><td>1126</td><td>Amy</td><td>King</td><td>195.84</td></tr>
<tr><td>5</td><td>1123</td><td>John</td><td>Doe</td><td>305.68</td></tr>
<tr><td>6</td><td>1128</td><td>Gordon</td><td>Knight</td><td>244.48</td></tr>
<tr><td>7</td><td>1129</td><td>Ted</td><td>Herta</td><td>282.64</td></tr>
<tr><td>8</td><td>1130</td><td>Amanda</td><td>Bryant</td><td>230.08</td></tr>
</table></td></tr>
</table>

In A5 we learned to create much more advanced queries and interesting queries. We were able to find more specific and isolated data. The first advanced query will display the combined employee details and document details, the second query will select all the jobs in the job table and separate the ones which do not have an assigned employee to it. The third query will check if there exists an employee whose annual salary is greater than 45 000, the fifth query will List all deductible amounts from table DEDUCTIBLES such that the PAYEE_NAME from table. Finally the sixth query will Retrieves the EMPLOYEE_ID, FIRST_NAME, LAST_NAME and sums all the deductible amounts over 100.

A6/A7/A8 - Normalization of the Database/Functional Dependencies/Normalization 3NF/Normalization BCNF

EMPLOYEE

BCNF/3NF	
Candidate Keys	Functional Dependencies
{EMPLOYEE_ID}	{EMPLOYEE_ID} → FIRST_NAME {EMPLOYEE_ID} → LAST_NAME {EMPLOYEE_ID} → DATE_OF_BIRTH {EMPLOYEE_ID} → SIN_NUM {EMPLOYEE_ID} → ADDRESS {EMPLOYEE_ID} → EMPLOYMENT_STATUS {EMPLOYEE_ID} → PERFORMANCE_BONUS
{EMPLOYEE_ID}$^+$ = {EMPLOYEE_ID, FIRST_NAME, LAST_NAME, DATE_OF_BIRTH, SIN_NUM, ADDRESS, EMPLOYMENT_STATUS, PERFORMANCE_BONUS}	

BANK_ACCOUNT

BCNF/3NF

Candidate Keys	Functional Dependencies
{ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID}	{ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID} → ACCOUNT_TYPE {ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID} → INSTITUTION_NO {ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID} → TRANSIT_NO {ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID} → ACCOUNT_NO
{ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID}$^+$ = {ACCOUNT_TYPE, INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO}	

JOB_TABLE

2NF	
Candidate Keys	Functional Dependencies
{JOB_ID, EMPLOYEE_ID}	{JOB_ID, EMPLOYEE_ID} → JOB_NAME {JOB_NAME} → JOB_DESC {JOB_ID, EMPLOYEE_ID} → BASE_PAY

BCNF/3NF	
Candidate Keys	Functional Dependencies
{JOB_ID, EMPLOYEE_ID}	{JOB_ID, EMPLOYEE_ID} → JOB_NAME

	{JOB_ID, EMPLOYEE_ID} → BASE_PAY
{JOB_ID, EMPLOYEE_ID}$^+$ = {JOB_ID, EMPLOYEE_ID, JOB_NAME, BASE_PAY}	
Candidate Keys	Functional Dependencies
{JOB_NAME}	{JOB_NAME} → JOB_DESC
{JOB_NAME}$^+$ = {JOB_NAME, JOB_DESC}	

SALARY

2NF	
Candidate Keys	Functional Dependencies
{EMPLOYEE_ID, JOB_ID, SALARY_ID}	{EMPLOYEE_ID, JOB_ID, SALARY_ID} → HOURLY_RATE {EMPLOYEE_ID, JOB_ID, SALARY_ID} → ANNUAL {EMPLOYEE_ID, JOB_ID, SALARY_ID} → ANNUAL_BONUS

BCNF/3NF

Candidate Keys	Functional Dependencies
{EMPLOYEE_ID, JOB_ID, SALARY_ID}	{EMPLOYEE_ID, JOB_ID, SALARY_ID} → HOURLY_RATE {EMPLOYEE_ID, JOB_ID, SALARY_ID} → ANNUAL_BONUS
{EMPLOYEE_ID, JOB_ID, SALARY_ID}$^{+}$ = { HOURLY_RATE, ANNUAL_BONUS}	

DEDUCTIBLES

1NF		
Primary Keys	Candidate Keys	Functional Dependencies
{DEDUCTIBLES_ID)	{DEDUCTIBLES_ID, JOB_ID, SALARY_ID}	{DEDUCTIBLES_ID, JOB_ID, SALARY_ID} → PAY_START {DEDUCTIBLES_ID, JOB_ID, SALARY_ID} → PAY_END {DEDUCTIBLES_ID, JOB_ID, SALARY_ID} → DEDUCTIBLE_RATE {DEDUCTIBLES_ID, JOB_ID, SALARY_ID} → DEDUCTIBLE_TYPE

		{DEDUCTIBLES_ID, JOB_ID, SALARY_ID} → DEDUCTIBLE_AMOUNT

BCNF	
Candidate Keys	Functional Dependencies
{DEDUCTIBLES_ID, SALARY_ID}	{DEDUCTIBLES_ID, SALARY_ID} → PAY_START {DEDUCTIBLES_ID, SALARY_ID} → PAY_END {DEDUCTIBLES_ID, SALARY_ID} → DEDUCTIBLE_TYPE
Candidate Keys	Functional Dependencies
{DEDUCTIBLE_TYPE}	{DEDUCTIBLE_TYPE} → DEDUCTIBLE_RATE

{DEDUCTIBLES_ID, SALARY_ID}$^+$ = { PAY_START, PAY_END, DEDUCTIBLE_TYPE}
{DEDUCTIBLE_TYPE}$^+$ = {DEDUCTIBLE_RATE}	

DEDUCTIBLES_PAYEE

BCNF/3NF	
Candidate Keys	Functional Dependencies
{DEDUCTIBLES_ID, PAYEE_ID}	{DEDUCTIBLES_ID, SALARY_ID} → PAYEE_NAME
{DEDUCTIBLES_ID, PAYEE_ID}$^+$ = {PAYEE_NAME}	

DOCUMENT_TABLE

BCNF/3NF	
Candidate Keys	Functional Dependencies
{DOC_ID, EMPLOYEE_ID}	{DOC_ID, EMPLOYEE_ID} → DOC_TYPE {DOC_ID, EMPLOYEE_ID} → DOC_NAME {DOC_ID, EMPLOYEE_ID} → ISSUE_DATE {DOC_ID, EMPLOYEE_ID} → EXPIRY_DATE
{DOC_ID, EMPLOYEE_ID}$^+$ = {DOC_TYPE, DOC_NAME, ISSUE_DATE, EXPIRY_DATE}	

PAYROLL

2NF	
Candidate Keys	Functional Dependencies
{PAYROLL_ID, SALARY_ID}	{PAYROLL_ID, SALARY_ID} → APPLICABLE_DATE {PAYROLL_ID, SALARY_ID} → PAYROLL_START {PAYROLL_ID, SALARY_ID} → PAYROLL_END {PAYROLL_ID, SALARY_ID} → PAYCODE {PAYROLL_ID, SALARY_ID} → HOURS_WORKED {PAYROLL_ID, SALARY_ID} → MULTIPLIER

BCNF/3NF	
Candidate Keys	Functional Dependencies
{PAYROLL_ID, SALARY_ID}	{PAYROLL_ID, SALARY_ID} → APPLICABLE_DATE {PAYROLL_ID, SALARY_ID} → PAYROLL_START

	{PAYROLL_ID, SALARY_ID} → PAYROLL_END {PAYROLL_ID, SALARY_ID} → PAYCODE {PAYROLL_ID, SALARY_ID} → HOURS_WORKED
Candidate Keys	Functional Dependencies
{PAYCODE}	{PAYCODE} → MULTIPLIER

{PAYROLL_ID, SALARY_ID}$^+$ = {APPLICABLE_DATE, PAYROLL_START, PAYROLL_END, PAYCODE, HOURS_WORKED}
{PAYCODE}$^+$ = {MULTIPLIER}	

Algorithmic Conversion from 2NF to BCNF/3NF

```
ATTRIBUTES $^+$ := STARTING ATTRIBUTES ;
repeat
        old ATTRIBUTES $^+$ := ATTRIBUTES $^+$;
        for each FunctionalDependency Y → Z in JOB_TABLE do
                If Y $\subseteq$ ATTRIBUTES $^+$
                Then ATTRIBUTES $^+$ := ATTRIBUTES $^+$ $\cup$ Z;
until (old ATTRIBUTES $^+$ = ATTRIBUTES $^+$);
```

ATTRIBUTES $^+$ = {JOB_NAME}

Y → Z {JOB_NAME} → JOB_DESC ATTRIBUTES $^{+}$ = {JOB_NAME, JOB_DESC}
ATTRIBUTES $^{+}$ = {JOB_ID, EMPLOYEE_ID} Y → Z {JOB_ID, EMPLOYEE_ID} → JOB_NAME ATTRIBUTES $^{+}$ = {JOB_NAME, EMPLOYEE_ID, JOB_NAME} Y → Z {JOB_ID, EMPLOYEE_ID} → BASE_PAY ATTRIBUTES $^{+}$ = {JOB_NAME, EMPLOYEE_ID, JOB_NAME, BASE_PAY}

Example Normalizations

Employee_Project

Attributes: EmployeeID, ProjectID, ProjectManagerID, ManagerOffice

Step 1: Determine functional dependencies

R (BookID, AuthorID, AuthorCountry)

FD = EmployeeID, ProjectID → ProjectManagerID, ManagerOffice

ProjectID → ProjectManagerID

ProjectManagerID → ManagerOffice

Step 2: Determine any candidate keys and violations

{EmployeeID, ProjectID}$^{+}$ = {EmployeeID, ProjectID, ProjectManagerID, ManagerOffice}

{EmployeeID, ProjectID} is a candidate key

{ProjectID}$^{+}$ = {ProjectID, ProjectManagerID, ManagerOffice}

{ProjectID} is not a candidate key, BCNF violation

{ProjectManagerID}$^+$ = {ProjectManagerID, ManagerOffice}

{ProjectManagerID} is not a candidate key, BCNF violation

Step 3: Decomposition of tables

{EmployeeID, ProjectID, ProjectManagerID, ManagerOffice} is not BCNF with respect to ProjectID → ProjectManagerID

Decompose into:

{EmployeeID, ProjectID} in BCNF

{ProjectID, ProjectManagerID, ManagerOffice}

{ProjectID, ProjectManagerID, ManagerOffice} is not BCNF with respect to ProjectManagerID → ManagerOffice

Decompose into:

{ProjectID, ProjectManagerID} in BCNF

{ProjectManagerID, ManagerOffice} in BCNF

4. Check for lossless join

[REmployeeAssignment(EmployeeID, ProjectID)] ∩ [RProjects(ProjectID, ProjectManagerID)] ∪ [RManagers(ProjectManagerID, ManagerOffice)]

5. Final Tables

{EmployeeID, ProjectID}

{ProjectID, ProjectManagerID}

{ProjectManagerID, ManagerOffice}

Book_Author

Attributes: BookID, AuthorID, AuthorCountry

1. Determine functional dependencies

R (BookID, AuthorID, AuthorCountry)

FD = {BookID → AuthorID
AuthorID → AuthorCountry}

2. Break RHS and find redundancies

BookID → AuthorID: BookID+ = {BookID, AuthorID}

AuthorID →AuthorCountry: AuthorID+ = {AuthorID, AuthorCountry}

Remove partial dependencies (Minimizing LHS)

BookID → AuthorID: BookID+ = {BookID, AuthorID}

3. Check for lossless join

[R_1(BookID, AuthorID) ∩ R_2(AuthorID, AuthorCountry)]

BookID, AuthorID+ = {BookID, AuthorID, AuthorCountry}

4. Make Tables

- In FD, BookID and AuthorID are on LHS therefore are part of the key.
- AuthorCountry is not on LHS and only on RHS therefore is NOT part of the key.

R_1 (BookID, AuthorID) with FD: BookID → AuthorID

R_2 (AuthorID, AuthorCountry) with FD: AuthorID → AuthorCountry

Final Tables:

{BookID, AuthorID}

{AuthorID, AuthorCountry}

Student_Class

Attributes: StudentID, ClassID, ClassRoom, ClassTime

1. Determine the functional dependencies

R(StudentID, ClassID, ClassRoom, ClassTime)

FD = {ClassID → ClassRoom, ClassTime

StudentID,ClassID → ClassRoom, ClassTime}

2. Identify candidate keys

{ClassID}+ = {ClassID, ClassRoom, ClassTime}

{ClassID} is not a candidate key.

{Student ID, ClassID}+ = {StudentID, ClassID, ClassRoom, ClassTime}

{Student ID, ClassID} is a candidate key.

3. Check for lossless join

[RClass(ClassID, ClassRoom, ClassTime)] ∩ [RStudent(StudentID, ClassID, ClassRoom, ClassTime)]

4. Final Decomposition

(StudentID, ClassID, ClassRoom, ClassTime)

Final Tables:

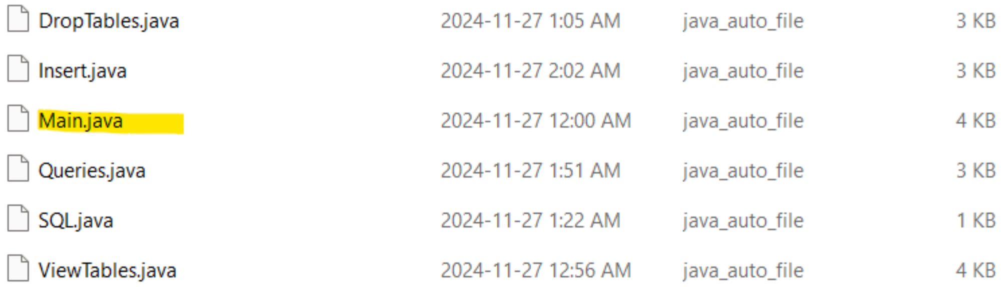
{ClassID, ClassRoom, ClassTime}

{StudentID, ClassID}

In A6 the goal was to create the tables and identify the functional dependencies. What we found was that most of our tables were already in 3NF/BCNF so A6 and A7 were done simultaneously. The goal of A8 was to normalize these tables further using Brenstiens algorithm into BCNF. By A8 we were already normalized to BCNF, so instead we chose three example normalizations to demonstrate our knowledge on Brensteins and algorithmic decomposition of tables. In this we also checked for lossless join, this was done by forming classes of functional dependencies and forming an intersection between them, this effectively keeps all the commonalities while removing redundancies and isolated attributes.

A9 - Demonstration of Application Using Java/web based GUI

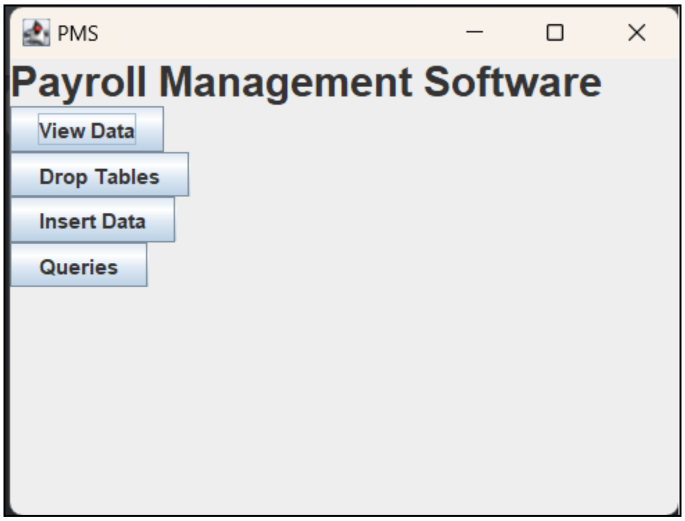
Setup



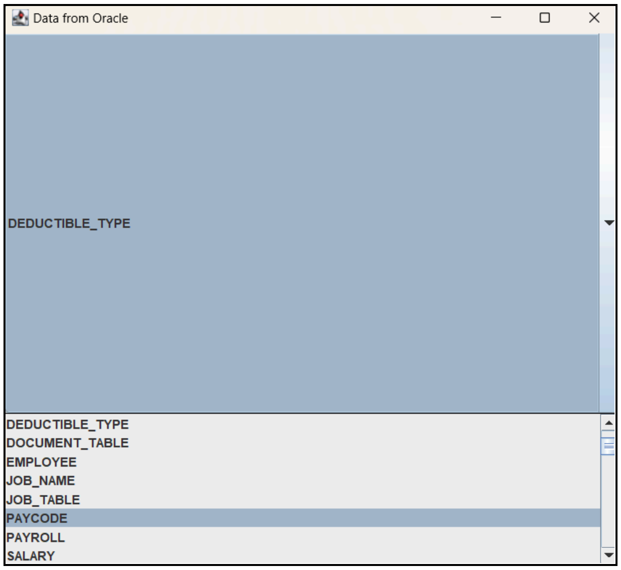
DropTables.java	2024-11-27 1:05 AM	java_auto_file	3 KB
Insert.java	2024-11-27 2:02 AM	java_auto_file	3 KB
Main.java	2024-11-27 12:00 AM	java_auto_file	4 KB
Queries.java	2024-11-27 1:51 AM	java_auto_file	3 KB
SQL.java	2024-11-27 1:22 AM	java_auto_file	1 KB
ViewTables.java	2024-11-27 12:56 AM	java_auto_file	4 KB

The GUI can be run through *Sec09_Team26_A09\src\Main.java* with JDK21 with JDBC dependencies. File *INSERTS.txt* includes data values that can be used as example values that can be used to populate the tables.

The Payroll Management System (PMS) DBMS has an embedded login within the code itself so that the application connects directly to the oracle database where the PMS is located.



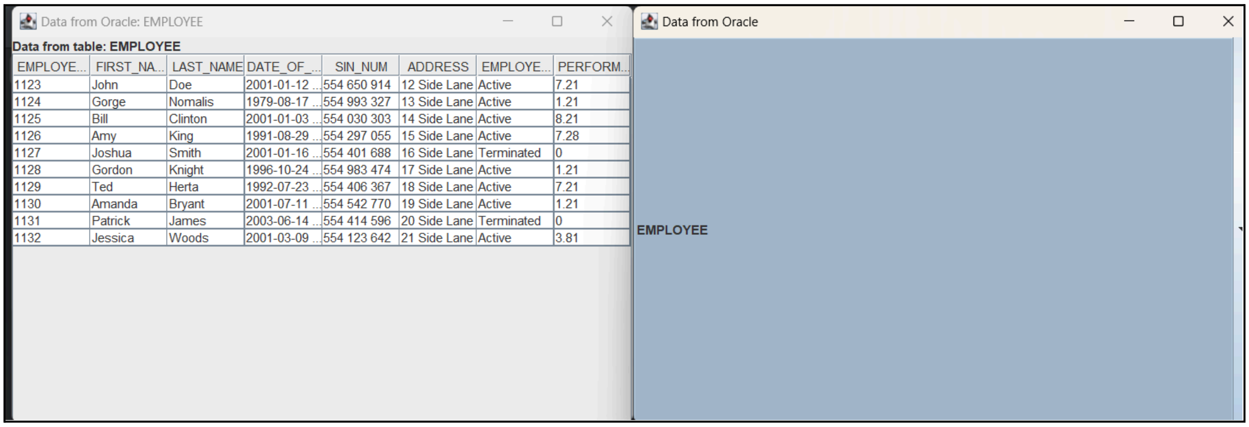
When the user is logged in, the following screen will appear prompting four options similar to the Unix implementation of the system.



DEDUCTIBLE_TYPE

DEDUCTIBLE_TYPE
DOCUMENT_TABLE
EMPLOYEE
JOB_NAME
JOB_TABLE
PAYCODE
PAYROLL
SALARY

View Data will display the following screen where the user can cycle through the various tables. The data is currently empty as data has not been inserted into the tables.

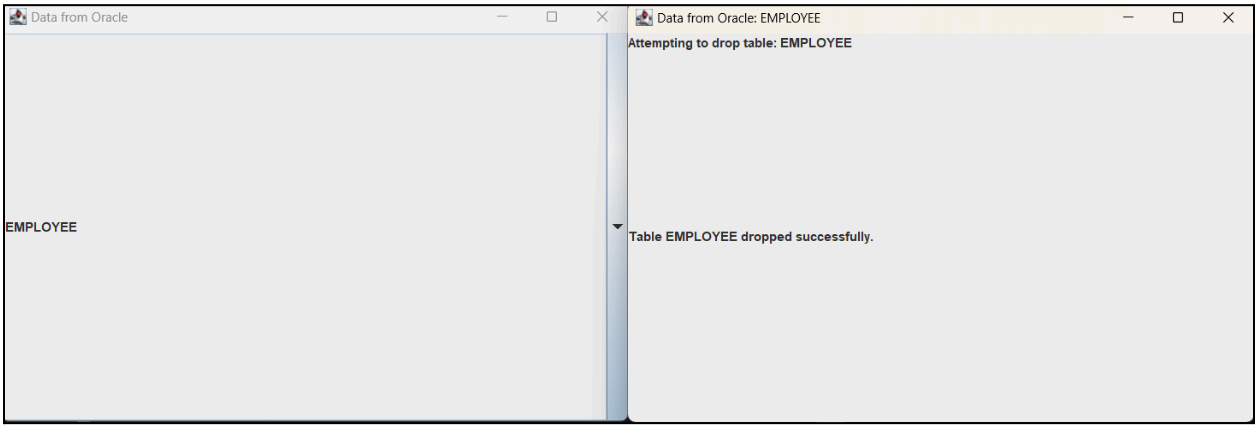


Data from table: EMPLOYEE

EMPLOYE...	FIRST_NA...	LAST_NAME	DATE_OF_...	SIN_NUM	ADDRESS	EMPLOYE...	PERFORM...
1123	John	Doe	2001-01-12 ...	554 650 914	12 Side Lane	Active	7.21
1124	Gorge	Nomalis	1979-08-17 ...	554 993 327	13 Side Lane	Active	1.21
1125	Bill	Clinton	2001-01-03 ...	554 030 303	14 Side Lane	Active	8.21
1126	Amy	King	1991-08-29 ...	554 297 055	15 Side Lane	Active	7.28
1127	Joshua	Smith	2001-01-16 ...	554 401 688	16 Side Lane	Terminated	0
1128	Gordon	Knight	1996-10-24 ...	554 983 474	17 Side Lane	Active	1.21
1129	Ted	Herta	1992-07-23 ...	554 406 367	18 Side Lane	Active	7.21
1130	Amanda	Bryant	2001-07-11 ...	554 542 770	19 Side Lane	Active	1.21
1131	Patrick	James	2003-06-14 ...	554 414 596	20 Side Lane	Terminated	0
1132	Jessica	Woods	2001-03-09 ...	554 123 642	21 Side Lane	Active	3.81

EMPLOYEE

When the tables are populated using the “Insert Data” option the data will appear in the table as it does within SQL developer, here the user can view each table in its original state with its respective attributes.



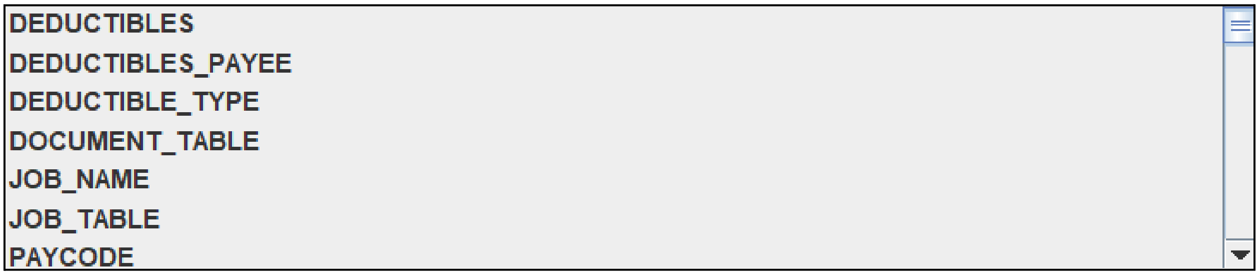
EMPLOYEE

Attempting to drop table: EMPLOYEE

Table EMPLOYEE dropped successfully.

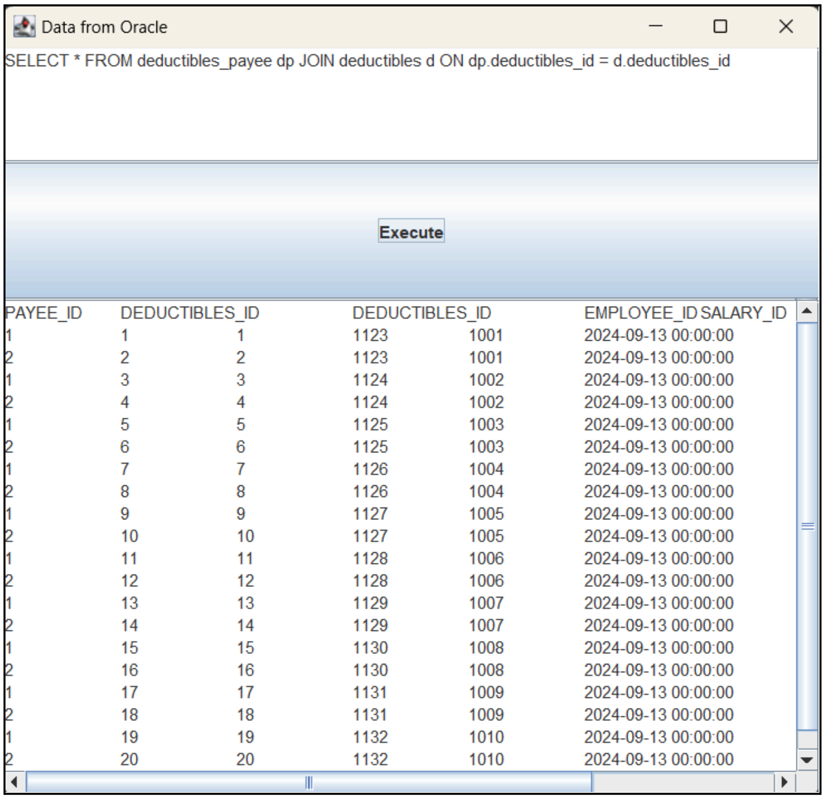
The “Drop Table” option will drop the selected table, where the user will receive a prompt confirming that the table has been dropped successfully.

(screenshot shows table EMPLOYEE has been dropped)



DEDUCTIBLES
DEDUCTIBLES_PAYEE
DEDUCTIBLE_TYPE
DOCUMENT_TABLE
JOB_NAME
JOB_TABLE
PAYCODE

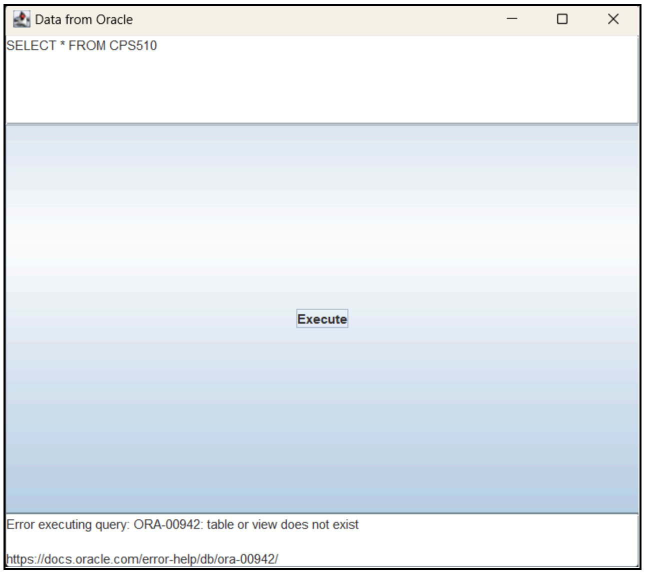
The table will then be removed from the selection of tables that can be viewed.



```
SELECT * FROM deductibles_payee dp JOIN deductibles d ON dp.deductibles_id = d.deductibles_id
```

```
PAYEE_ID    DEDUCTIBLES_ID          DEDUCTIBLES_ID          EMPLOYEE_ID SALARY_ID
1           1           1           1123        1001        2024-09-13 00:00:00
2           2           2           1123        1001        2024-09-13 00:00:00
1           3           3           1124        1002        2024-09-13 00:00:00
2           4           4           1124        1002        2024-09-13 00:00:00
1           5           5           1125        1003        2024-09-13 00:00:00
2           6           6           1125        1003        2024-09-13 00:00:00
1           7           7           1126        1004        2024-09-13 00:00:00
2           8           8           1126        1004        2024-09-13 00:00:00
1           9           9           1127        1005        2024-09-13 00:00:00
2           10          10          1127        1005        2024-09-13 00:00:00
1           11          11          1128        1006        2024-09-13 00:00:00
2           12          12          1128        1006        2024-09-13 00:00:00
1           13          13          1129        1007        2024-09-13 00:00:00
2           14          14          1129        1007        2024-09-13 00:00:00
1           15          15          1130        1008        2024-09-13 00:00:00
2           16          16          1130        1008        2024-09-13 00:00:00
1           17          17          1131        1009        2024-09-13 00:00:00
2           18          18          1131        1009        2024-09-13 00:00:00
1           19          19          1132        1010        2024-09-13 00:00:00
2           20          20          1132        1010        2024-09-13 00:00:00
```

Using the “Queries” option the user is prompted with a textbox. An SQL query can then be manually entered into the textbox and clicking the execute command will display the query entered. Otherwise an error message will appear at the bottom of the application.



```
SELECT * FROM CPS510
```

```
Error executing query: ORA-00942: table or view does not exist

https://docs.oracle.com/error-help/db/ora-00942/
```

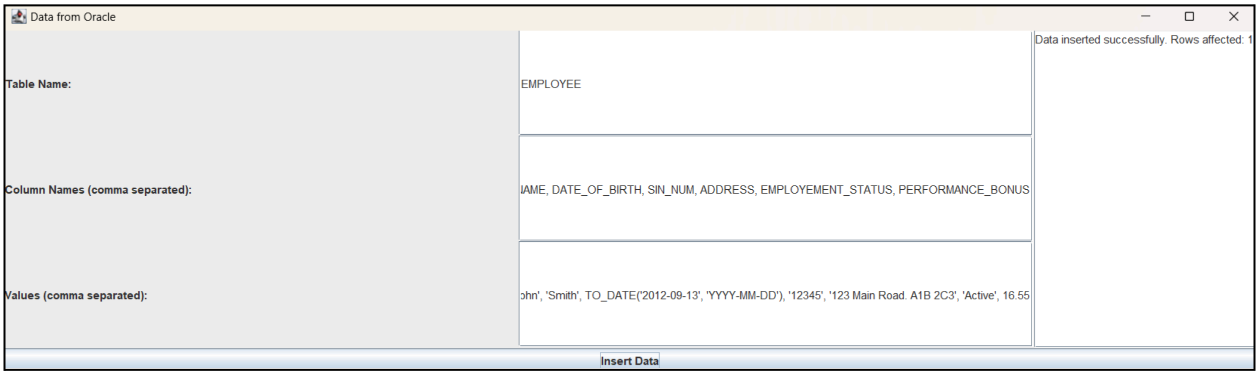


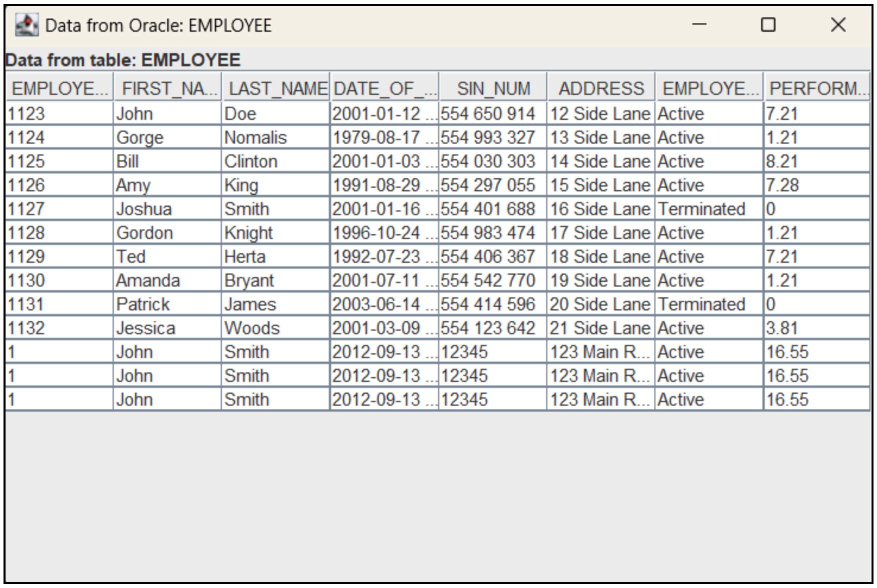
Table Name: EMPLOYEE

Column Names (comma separated): IAME, DATE_OF_BIRTH, SIN_NUM, ADDRESS, EMPLOYEMENT_STATUS, PERFORMANCE_BONUS

Values (comma separated): ohn', 'Smith', TO_DATE('2012-09-13', 'YYYY-MM-DD'), '12345', '123 Main Road. A1B 2C3', 'Active', 16.55

```
Data inserted successfully. Rows affected: 1
```

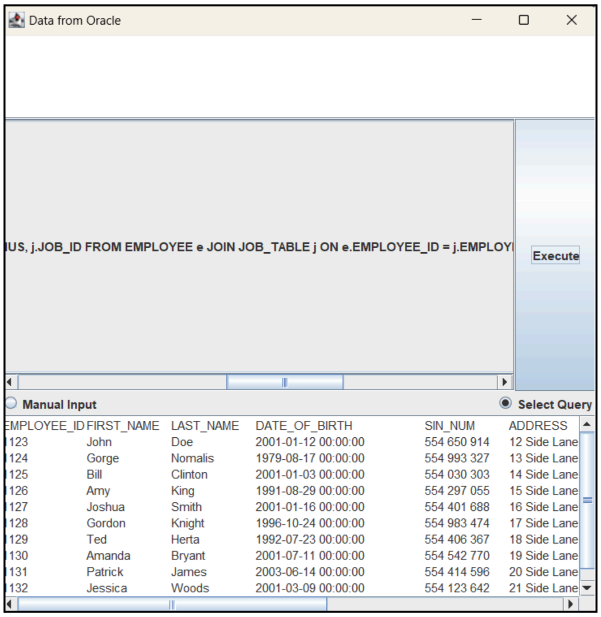
Using the “Insert Data” option the user is prompted with several text boxes. The information for the table can then be manually entered in, using a comma to separate the column names and values. This data can then be inserted into the database as a table.



Data from table: EMPLOYEE

EMPLOYE...	FIRST_NA...	LAST_NAME	DATE_OF_...	SIN_NUM	ADDRESS	EMPLOYE...	PERFORM...
1123	John	Doe	2001-01-12 ...	554 650 914	12 Side Lane	Active	7.21
1124	Gorge	Nomalis	1979-08-17 ...	554 993 327	13 Side Lane	Active	1.21
1125	Bill	Clinton	2001-01-03 ...	554 030 303	14 Side Lane	Active	8.21
1126	Amy	King	1991-08-29 ...	554 297 055	15 Side Lane	Active	7.28
1127	Joshua	Smith	2001-01-16 ...	554 401 688	16 Side Lane	Terminated	0
1128	Gordon	Knight	1996-10-24 ...	554 983 474	17 Side Lane	Active	1.21
1129	Ted	Herta	1992-07-23 ...	554 406 367	18 Side Lane	Active	7.21
1130	Amanda	Bryant	2001-07-11 ...	554 542 770	19 Side Lane	Active	1.21
1131	Patrick	James	2003-06-14 ...	554 414 596	20 Side Lane	Terminated	0
1132	Jessica	Woods	2001-03-09 ...	554 123 642	21 Side Lane	Active	3.81
1	John	Smith	2012-09-13 ...	12345	123 Main R...	Active	16.55
1	John	Smith	2012-09-13 ...	12345	123 Main R...	Active	16.55
1	John	Smith	2012-09-13 ...	12345	123 Main R...	Active	16.55

Once inserted into the database the table can be viewed using the “View Table” option or interacted with any of the other options.



```
US, j.JOB_ID FROM EMPLOYEE e JOIN JOB_TABLE j ON e.EMPLOYEE_ID = j.EMPLOYI
```

○ Manual Input ● Select Query

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DATE_OF_BIRTH	SIN_NUM	ADDRESS
1123	John	Doe	2001-01-12 00:00:00	554 650 914	12 Side Lane
1124	Gorge	Nomalis	1979-08-17 00:00:00	554 993 327	13 Side Lane
1125	Bill	Clinton	2001-01-03 00:00:00	554 030 303	14 Side Lane
1126	Amy	King	1991-08-29 00:00:00	554 297 055	15 Side Lane
1127	Joshua	Smith	2001-01-16 00:00:00	554 401 688	16 Side Lane
1128	Gordon	Knight	1996-10-24 00:00:00	554 983 474	17 Side Lane
1129	Ted	Herta	1992-07-23 00:00:00	554 406 367	18 Side Lane
1130	Amanda	Bryant	2001-07-11 00:00:00	554 542 770	19 Side Lane
1131	Patrick	James	2003-06-14 00:00:00	554 414 596	20 Side Lane
1132	Jessica	Woods	2001-03-09 00:00:00	554 123 642	21 Side Lane

Alternatively, queries can be selected from a list, in this image the 'Select Query' button is activated and a query is chosen from the list. The query used here was advance query join: `SELECT e.EMPLOYEE_ID, e.FIRST_NAME, e.LAST_NAME, e.DATE_OF_BIRTH, e.SIN_NUM, e.ADDRESS, e.EMPLOYEMENT_STATUS, e.PERFORMANCE_BONUS, j.JOB_ID FROM EMPLOYEE e JOIN JOB_TABLE j ON e.EMPLOYEE_ID = j.EMPLOYEE_ID`

Appendix - A3

```
DROP TABLE PAYROLL;
DROP TABLE document_table;
DROP TABLE bank_account;
DROP TABLE Deductibles_payee;
DROP TABLE deductibles;
DROP TABLE salary;
DROP TABLE job_table;
DROP TABLE employee;

CREATE TABLE EMPLOYEE
(
   EMPLOYEE_ID INT PRIMARY KEY NOT NULL,
   FIRST_NAME VARCHAR(20),
   LAST_NAME VARCHAR(20),
   DATE_OF_BIRTH DATE,
   SIN_NUM VARCHAR(30),
   ADDRESS VARCHAR(50),
   EMPLOYMENT_STATUS VARCHAR(20),
   PERFORMANCE_BONUS FLOAT
);

CREATE TABLE BANK_ACCOUNT
(
   ACCOUNT_ID INT,
   EMPLOYEE_ID INT,
   PAYEE_ID INT,
   ACCOUNT_TYPE INT,
   INSTITUTION_NO INT,
   TRANSIT_NO INT,
    ACCOUNT_NO INT,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);

CREATE TABLE JOB_TABLE
(
    JOB_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    JOB_NAME VARCHAR(100),
    JOB_DESC VARCHAR(100),
    BASE_PAY FLOAT,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);

CREATE TABLE SALARY
(
    SALARY_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    JOB_ID INT,
    HOURLY_RATE FLOAT,
    ANNUAL FLOAT,
    ANNUAL_BONUS FLOAT,
    FOREIGN KEY (JOB_ID) REFERENCES job_table(JOB_ID),
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);
CREATE TABLE DEDUCTIBLES
(
    DEDUCTIBLES_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    SALARY_ID INT,
    PAY_START DATE,
    PAY_END DATE,
    DEDUCTIBLE_RATE FLOAT,
    DEDUCTIBLE_TYPE VARCHAR(100),
    DEDUCTIBLE_AMOUNT FLOAT,
    FOREIGN KEY (SALARY_ID) REFERENCES salary(SALARY_ID),
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);

CREATE TABLE DEDUCTIBLES_PAYEE
(
    PAYEE_ID INT PRIMARY KEY NOT NULL,
    DEDUCTIBLES_ID INT,
    PAYEE_NAME VARCHAR(20),
    FOREIGN KEY (DEDUCTIBLES_ID) REFERENCES deductibles(DEDUCTIBLES_ID)
);


CREATE TABLE DOCUMENT_TABLE
(
    DOC_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    DOC_TYPE VARCHAR(30),
    DOC_NAME VARCHAR(30),
    ISSUE_DATE DATE,
    EXPIRY_DATE DATE,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);


CREATE TABLE PAYROLL
(
    PAYROLL_ID INT PRIMARY KEY NOT NULL,
    SALARY_ID INT,
    DEDUCTIBLES_ID INT,
    APPLICABLE_DATE DATE,
    PAYROLL_START DATE,
    PAYROLL_END DATE,
    PAYCODE VARCHAR(10),
    HOURS_WORKED INT,
    MULTIPLIER FLOAT,
    FOREIGN KEY (SALARY_ID) REFERENCES salary(SALARY_ID),
    FOREIGN KEY (DEDUCTIBLES_ID) REFERENCES deductibles(DEDUCTIBLES_ID)
);
//
INSERT INTO EMPLOYEE (EMPLOYEE_ID, FIRST_NAME, LAST_NAME,
DATE_OF_BIRTH, SIN_NUM, ADDRESS, EMPLOYMENT_STATUS,
PERFORMANCE_BONUS)
VALUES
(1, 'John', 'Smith', TO_DATE('2012-09-13', 'YYYY-MM-DD'), '12345', '123 Main Road. A1B
2C3', 'Active', 16.55);
INSERT INTO EMPLOYEE (EMPLOYEE_ID, FIRST_NAME, LAST_NAME,
DATE_OF_BIRTH, SIN_NUM, ADDRESS, EMPLOYMENT_STATUS,
PERFORMANCE_BONUS)
VALUES
(2, 'Bob', 'Doe', TO_DATE('2009-12-31', 'YYYY-MM-DD'), '23456', '234 Side Road. D4E 5F6',
'Terminated', 0.00);

INSERT INTO BANK_ACCOUNT (ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID,
ACCOUNT_TYPE, INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO)
VALUES
(1, 1, 123, 456, 11111);
INSERT INTO BANK_ACCOUNT (ACCOUNT_ID, EMPLOYEE_ID,PAYEE_ID,
ACCOUNT_TYPE, INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO)
VALUES
(2, 2, 321, 654, 22222);


INSERT INTO JOB_TABLE (JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC, BASE_PAY)
VALUES
```

```
(1, 1, 'Manager', 'Manages team', 16.55);
INSERT INTO JOB_TABLE (JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC, BASE_PAY)
VALUES
(2, 2, 'Secretary', 'Office Management', 17.20);


INSERT INTO SALARY (SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE, ANNUAL,
ANNUAL_BONUS)
VALUES
(1, 1, 1, 10000, 10000, 2000);
INSERT INTO SALARY (SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE, ANNUAL,
ANNUAL_BONUS)
VALUES
(2, 2, 2, 11000, 11000, 1000);

INSERT INTO DEDUCTIBLES (DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID,
PAY_START, PAY_END, DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE,
DEDUCTIBLE_AMOUNT)
VALUES
(1, 1, 1,TO_DATE('2024-09-13', 'YYYY-MM-DD'), TO_DATE('2024-09-27', 'YYYY-MM-DD'),
0.1, 'Tax', 4789.00 );
INSERT INTO DEDUCTIBLES (DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID,
PAY_START, PAY_END, DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE,
DEDUCTIBLE_AMOUNT)
VALUES
(2, 2, 2, TO_DATE('2024-09-13', 'YYYY-MM-DD'), TO_DATE('2024-09-27', 'YYYY-MM-DD'),
0.2, 'Insurance', 5987.00 );

INSERT INTO DOCUMENT_TABLE (DOC_ID, EMPLOYEE_ID, DOC_TYPE, DOC_NAME,
ISSUE_DATE, EXPIRY_DATE) VALUES
(1, 1, 'Employment Contract', 'Contract_2024ver', TO_DATE('2024-09-13','YYYY-MM-DD'),
TO_DATE('2024-09-14', 'YYYY-MM-DD'));

INSERT INTO DOCUMENT_TABLE (DOC_ID, EMPLOYEE_ID, DOC_TYPE, DOC_NAME,
ISSUE_DATE, EXPIRY_DATE) VALUES
(2, 2, 'Identification', 'Driver’s License', TO_DATE('2024-09-12','YYYY-MM-DD'),
TO_DATE('2024-09-13', 'YYYY-MM-DD'));

INSERT INTO PAYROLL (PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID,
APPLICABLE_DATE, PAYROLL_START, PAYROLL_END, PAYCODE, HOURS_WORKED,
MULTIPLIER)
VALUES
(1, 1, 1, TO_DATE('2024-10-04', 'YYYY-MM-DD'), TO_DATE('2024-09-13', 'YYYY-MM-DD'),
TO_DATE('2024-09-27', 'YYYY-MM-DD'), 'regular', 80, 1.0);

INSERT INTO PAYROLL (PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID,
APPLICABLE_DATE, PAYROLL_START, PAYROLL_END, PAYCODE, HOURS_WORKED,
MULTIPLIER)
VALUES
(2, 2, 2, TO_DATE('2024-10-04', 'YYYY-MM-DD'), TO_DATE('2024-09-13', 'YYYY-MM-DD'),
TO_DATE('2024-09-27', 'YYYY-MM-DD'), 'stat', 65, 2.0);

SELECT
   EMPLOYEE_ID, FIRST_NAME, LAST_NAME, DATE_OF_BIRTH, SIN_NUM, ADDRESS,
EMPLOYMENT_STATUS, PERFORMANCE_BONUS
FROM
   EMPLOYEE
WHERE
   EMPLOYEE_ID = 1;

SELECT
   ACCOUNT_ID, EMPLOYEE_ID,PAYEE_ID, ACCOUNT_TYPE,  INSTITUTION_NO,
TRANSIT_NO, ACCOUNT_NO
FROM
   BANK_ACCOUNT
WHERE
   ACCOUNT_ID = 1 AND ACCOUNT_NO = 11111;

SELECT
   JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC, BASE_PAY
FROM
   JOB_TABLE
WHERE
   EMPLOYEE_ID = 1 AND JOB_ID = 1;

SELECT
   SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE, ANNUAL, BONUS
FROM
   SALARY
WHERE
   HOURLY_RATE= 10000 AND SALARY_ID = 1;

SELECT
   DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID, PAY_START, PAY_END,
DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE, DEDUCTIBLE_AMOUNT
FROM
   DEDUCTIBLES
WHERE
   DEDUCTIBLES_ID = 1;

SELECT
   PAYEE_ID INT, DEDUCTIBLES_ID, PAYEE_NAME
FROM
   DEDUCTIBLES_PAYEE
WHERE
   PAYEE_ID = 101 AND DEDUCTIBLES_ID = 1;

SELECT
   DOC_ID, EMPLOYEE_ID, DOC_TYPE, DOC_NAME, ISSUE_DATE, EXPIRY_DATE
FROM
```

```
   DOCUMENT_TABLE
WHERE
   EMPLOYEE_ID = 1;

SELECT
   PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID, APPLICABLE_DATE, PAYROLL_START,
PAYROLL_END, PAYCODE, HOURS_WORKED, MULTIPLIER
FROM
   PAYROLL
WHERE
   PAYROLL_ID = 1 AND HOURS_WORKED > 0;
```

Appendix - A5

mainmenu.sh

```
#!/bin/sh
MainMenu()
{

 while [ "$CHOICE" != "START" ]
 do
 clear
 echo
"==============================================================
="
 echo "| Welcome to the Payroll DBMS
|"
echo "| Manage Employees Payroll, Documents and Information
|"
 echo "| Main Menu - Select Desired Operation(s):
|"
 echo "| <CTRL-Z Anytime to Enter Interactive CMD Prompt>
|"
 echo "------------------------------------------------------------
----"
 echo " $IS_SELECTEDM M) View Manual"
 echo " "
 echo " $IS_SELECTED1 1) Drop Tables"
 echo " $IS_SELECTED2 2) Create Tables"
 echo " $IS_SELECTED3 3) Populate Tables"
 echo " $IS_SELECTED4 4) Query Tables"
 echo " "
 echo " $IS_SELECTEDX X) Force/Stop/Kill Oracle DB"
 echo " "
 echo " $IS_SELECTEDE E) End/Exit"
 echo "Choose: "
 read CHOICE
 if [ "$CHOICE" == "0" ]
 then
 echo "Nothing Here"
 elif [ "$CHOICE" == "1" ]
 then
 bash drop_tables.sh
  read -n 1 -s
 elif [ "$CHOICE" == "2" ]
 then
 bash create_tables.sh
  read -n 1 -s
 elif [ "$CHOICE" == "3" ]
 then
 bash populate_tables.sh
  read -n 1 -s
 elif [ "$CHOICE" == "4" ]
 then
 bash queries_select.sh
 read -n 1 -s

 elif [ "$CHOICE" == "E" ]
 then
 exit
 fi
 done
}
#--COMMENTS BLOCK--
: 'mainmenu.sh’ extends files drop_tables.sh, create_tables.sh, populate_tables.sh,
and queries.sh with options 1,2,3,4 respectively.
Choosing the option runs the files with the queries included in each respective file.

'


#--COMMENTS BLOCK--
ProgramStart()
{
 StartMessage
 while [ 1 ]
 do
 MainMenu
 done
}
ProgramStart()
```

Drop_tables.sh (1)

```
#!/bin/sh
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib

MainMenu()
{

 while [ "$CHOICE" != "START" ]
 do
 clear

echo"==================================================================
===="
 echo "| Oracle All Inclusive Tool
|"
 echo "| Main Menu - Select Desired Operation(s):
|"
 echo "| <CTRL-Z Anytime to Enter Interactive CMD Prompt>
|"
 echo "------------------------------------------------------------------
----"
 echo " Select Table to Drop"
 echo " $IS_SELECTED1 1) Payroll"
 echo " $IS_SELECTED2 2) Documents"
 echo " $IS_SELECTED3 3) Bank Accounts"
 echo " $IS_SELECTED4 4) Deductibles payee"
 echo " $IS_SELECTED1 5) Deductibles"
 echo " $IS_SELECTED2 6) Salary"
 echo " $IS_SELECTED3 7) Jobs"
 echo " $IS_SELECTED4 8) Employee"
 echo " $IS_SELECTEDX X) Force/Stop/Kill Oracle DB"
 echo " "
 echo " $IS_SELECTEDE E) End/Exit"
 echo "Choose: "
 read CHOICE
 if [ "$CHOICE" == "0" ]
 then
 echo "Nothing Here"
echo “Press any key to continue…”
read -n 1 -s
 elif [ "$CHOICE" == "1" ]
 then

sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

DROP TABLE PAYROLL;

exit;
EOF
echo “Press any key to continue…”
 read -n 1 -s

 elif [ "$CHOICE" == "2" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

DROP TABLE document_table;

exit;
EOF
echo “Press any key to continue…”
read -n 1 -s
 elif [ "$CHOICE" == "3" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

DROP TABLE bank_account;

exit;
EOF
  echo “Press any key to continue…”
  read -n 1 -s
 elif [ "$CHOICE" == "4" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
```

```
DROP TABLE Deductibles_payee;

exit;
EOF
echo “Press any key to continue…”
 read -n 1 -s
 elif [ "$CHOICE" == "5" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

DROP TABLE deductibles;

exit;
EOF
echo “Press any key to continue…”
 read -n 1 -s
 elif [ "$CHOICE" == "6" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

DROP TABLE salary;

exit;
EOF
echo “Press any key to continue…”
  read -n 1 -s
 elif [ "$CHOICE" == "7" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

DROP TABLE job_table;

exit;
EOF
echo “Press any key to continue…”
  read -n 1 -s
 elif [ "$CHOICE" == "8" ]
 then
sqlplus64
```

```
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

DROP TABLE employee;

exit;
EOF
echo “Press any key to continue…”
 read -n 1 -s

 elif [ "$CHOICE" == "E" ]
 then
 exit
 fi
 done
}

#--COMMENTS BLOCK--
ProgramStart()
{
 StartMessage
 while [ 1 ]
 do
 MainMenu

 done
}
ProgramStart
```

Create_table.sh (2)

```
#!/bin/sh
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib
MainMenu()
{

 while [ "$CHOICE" != "START" ]
 do
 clear
 echo
"==============================================================
="
 echo "| Oracle All Inclusive Tool|"
 echo "| Main Menu - Select Desired Operation(s):|"
 echo "| <CTRL-Z Anytime to Enter Interactive CMD Prompt>|"
 echo "--------------------------------------------------------------"
 echo " Create a table:  "
 echo " $IS_SELECTED1 1) EMPLOYEE"
 echo " $IS_SELECTED2 2) BANK_ACCOUNT"
 echo " $IS_SELECTED3 3) JOB_TABLE"
 echo " $IS_SELECTED4 4) SALARY"
 echo " $IS_SELECTED5 5) DEDUCTIBLES"
 echo " $IS_SELECTED6 6) DEDUCTIBLES_PAYEE"
 echo " $IS_SELECTED7 7) DOCUMENT_TABLE"
 echo " $IS_SELECTED8 8) PAYROLL"
 echo " $IS_SELECTEDa a) ALL"
 echo " $IS_SELECTEDX X) Force/Stop/Kill Oracle DB"
 echo " "
 echo " $IS_SELECTEDE E) End/Exit"
 echo "Choose: "
 read CHOICE
 if [ "$CHOICE" == "0" ]
 then
 echo "Nothing Here"
 elif [ "$CHOICE" == "1" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE EMPLOYEE
(
    EMPLOYEE_ID INT PRIMARY KEY NOT NULL,
    FIRST_NAME VARCHAR(20),
    LAST_NAME VARCHAR(20),
    DATE_OF_BIRTH DATE,
    SIN_NUM VARCHAR(30),
    ADDRESS VARCHAR(50),
    EMPLOYMENT_STATUS VARCHAR(20),
    PERFORMANCE_BONUS FLOAT
);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "2" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE BANK_ACCOUNT
(
    ACCOUNT_ID INT,
    EMPLOYEE_ID INT,
    PAYEE_ID INT,
    ACCOUNT_TYPE VARCHAR(20),
    INSTITUTION_NO INT,
    TRANSIT_NO INT,
    ACCOUNT_NO INT,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "3" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE JOB_TABLE
(
    JOB_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    JOB_NAME VARCHAR(100),
    JOB_DESC VARCHAR(100),
    BASE_PAY FLOAT,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "4" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE SALARY
(
    SALARY_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    JOB_ID INT,
    HOURLY_RATE FLOAT,
    ANNUAL FLOAT,
    ANNUAL_BONUS FLOAT,
    FOREIGN KEY (JOB_ID) REFERENCES job_table(JOB_ID),
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);
exit;
```

```
EOF
 read -n 1 -s
 elif [ "$CHOICE" == "5" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE DEDUCTIBLES
(
    DEDUCTIBLES_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    SALARY_ID INT,
    PAY_START DATE,
    PAY_END DATE,
    DEDUCTIBLE_RATE FLOAT,
    DEDUCTIBLE_TYPE VARCHAR(100),
    DEDUCTIBLE_AMOUNT FLOAT,
    FOREIGN KEY (SALARY_ID) REFERENCES salary(SALARY_ID),
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "6" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE DEDUCTIBLES_PAYEE
(
    PAYEE_ID INT PRIMARY KEY NOT NULL,
    DEDUCTIBLES_ID INT,
    PAYEE_NAME VARCHAR(20),
    FOREIGN KEY (DEDUCTIBLES_ID) REFERENCES
deductibles(DEDUCTIBLES_ID)
);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "7" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE DOCUMENT_TABLE
(
    DOC_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    DOC_TYPE VARCHAR(30),
    DOC_NAME VARCHAR(30),
    ISSUE_DATE DATE,
    EXPIRY_DATE DATE,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "8" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE PAYROLL
(
    PAYROLL_ID INT PRIMARY KEY NOT NULL,
    SALARY_ID INT,
    DEDUCTIBLES_ID INT,
    APPLICABLE_DATE DATE,
    PAYROLL_START DATE,
    PAYROLL_END DATE,
    PAYCODE VARCHAR(10),
    HOURS_WORKED INT,
    MULTIPLIER FLOAT,
    FOREIGN KEY (SALARY_ID) REFERENCES salary(SALARY_ID),
    FOREIGN KEY (DEDUCTIBLES_ID) REFERENCES
deductibles(DEDUCTIBLES_ID)
);
exit;
EOF
 read -n 1 -s
 elif [ "$CHOICE" == "a" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
CREATE TABLE EMPLOYEE
(
    EMPLOYEE_ID INT PRIMARY KEY NOT NULL,
    FIRST_NAME VARCHAR(20),
    LAST_NAME VARCHAR(20),
    DATE_OF_BIRTH DATE,
    SIN_NUM VARCHAR(30),
    ADDRESS VARCHAR(50),
    EMPLOYMENT_STATUS VARCHAR(20),
    PERFORMANCE_BONUS FLOAT
);

CREATE TABLE BANK_ACCOUNT
(
    ACCOUNT_ID INT,
    EMPLOYEE_ID INT,
    PAYEE_ID INT,
    ACCOUNT_TYPE VARCHAR(20),
    INSTITUTION_NO INT,
    TRANSIT_NO INT,
    ACCOUNT_NO INT,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);

CREATE TABLE JOB_TABLE
(
    JOB_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    JOB_NAME VARCHAR(100),
    JOB_DESC VARCHAR(100),
    BASE_PAY FLOAT,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);

CREATE TABLE SALARY
(
    SALARY_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    JOB_ID INT,
    HOURLY_RATE FLOAT,
    ANNUAL FLOAT,
    ANNUAL_BONUS FLOAT,
    FOREIGN KEY (JOB_ID) REFERENCES job_table(JOB_ID),
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);
CREATE TABLE DEDUCTIBLES
(
    DEDUCTIBLES_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    SALARY_ID INT,
    PAY_START DATE,
    PAY_END DATE,
    DEDUCTIBLE_RATE FLOAT,
    DEDUCTIBLE_TYPE VARCHAR(100),
    DEDUCTIBLE_AMOUNT FLOAT,
    FOREIGN KEY (SALARY_ID) REFERENCES salary(SALARY_ID),
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);

CREATE TABLE DEDUCTIBLES_PAYEE
(
    PAYEE_ID INT PRIMARY KEY NOT NULL,
    DEDUCTIBLES_ID INT,
    PAYEE_NAME VARCHAR(20),
    FOREIGN KEY (DEDUCTIBLES_ID) REFERENCES
deductibles(DEDUCTIBLES_ID)
);


CREATE TABLE DOCUMENT_TABLE
(
    DOC_ID INT PRIMARY KEY NOT NULL,
    EMPLOYEE_ID INT,
    DOC_TYPE VARCHAR(30),
    DOC_NAME VARCHAR(30),
    ISSUE_DATE DATE,
    EXPIRY_DATE DATE,
    FOREIGN KEY (EMPLOYEE_ID) REFERENCES employee(EMPLOYEE_ID)
);


CREATE TABLE PAYROLL
(
    PAYROLL_ID INT PRIMARY KEY NOT NULL,
    SALARY_ID INT,
    DEDUCTIBLES_ID INT,
    APPLICABLE_DATE DATE,
    PAYROLL_START DATE,
    PAYROLL_END DATE,
    PAYCODE VARCHAR(10),
    HOURS_WORKED INT,
    MULTIPLIER FLOAT,
    FOREIGN KEY (SALARY_ID) REFERENCES salary(SALARY_ID),
    FOREIGN KEY (DEDUCTIBLES_ID) REFERENCES
deductibles(DEDUCTIBLES_ID)
);
exit;
EOF
  read -n 1 -s
```

```
 elif [ "$CHOICE" == "E" ]
 then
 exit
 fi
 done
}


#--COMMENTS BLOCK--
ProgramStart()
{
 StartMessage
 while [ 1 ]
 do
 MainMenu
 done
}
ProgramStart
```

Populate_tables.sh (3)

```
#!/bin/sh
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib

MainMenu()
{

 while [ "$CHOICE" != "START" ]
 do
 clear
 echo
"================================================================
="
 echo "| Oracle All Inclusive Tool
|"
 echo "| Main Menu - Select Desired Operation(s):
|"
 echo "| <CTRL-Z Anytime to Enter Interactive CMD Prompt>
|"
 echo "-------------------------------------------------------------
----"
 echo " Insert Into:"
 echo " $IS_SELECTED1 1) EMPLOYEE"
 echo " $IS_SELECTED2 2) EMPLOYEE"
 echo " $IS_SELECTED3 3) BANK_ACCOUNT"
 echo " $IS_SELECTED4 4) BANK_ACCOUNT"
 echo " $IS_SELECTED5 5) JOB_TABLE"
 echo " $IS_SELECTED6 6) JOB_TABLE"
 echo " $IS_SELECTED7 7) SALARY"
 echo " $IS_SELECTED8 8) SALARY"
 echo " $IS_SELECTED9 9) DEDUCTIBLES"
 echo " $IS_SELECTED10 10) DEDUCTIBLES"
 echo " $IS_SELECTED11 11) DOCUMENT_TABLE"
 echo " $IS_SELECTED12 12) DOCUMENT_TABLE"
 echo " $IS_SELECTED13 13) PAYROLL"
 echo " $IS_SELECTED14 14) PAYROLL"
 echo " $IS_SELECTED4 a) ALL"
 echo " $IS_SELECTEDX X) Force/Stop/Kill Oracle DB"
 echo " "
 echo " $IS_SELECTEDE E) End/Exit"
 echo "Choose: "
 read CHOICE
 if [ "$CHOICE" == "0" ]
 then
 echo "Nothing Here"
 elif [ "$CHOICE" == "1" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
INSERT INTO EMPLOYEE (EMPLOYEE_ID, FIRST_NAME, LAST_NAME,
DATE_OF_BIRTH, SIN_NUM, ADDRESS, EMPLOYMENT_STATUS,
PERFORMANCE_BONUS)
VALUES
(1, 'John', 'Smith', TO_DATE('2012-09-13', 'YYYY-MM-DD'), '12345', '123 Main Road.
A1B 2C3', 'Active', 16.55);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "2" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
INSERT INTO EMPLOYEE (EMPLOYEE_ID, FIRST_NAME, LAST_NAME,
DATE_OF_BIRTH, SIN_NUM, ADDRESS, EMPLOYMENT_STATUS,
PERFORMANCE_BONUS)
VALUES
```

```
(2, 'Bob', 'Doe', TO_DATE('2009-12-31', 'YYYY-MM-DD'), '23456', '234 Side Road.
D4E 5F6', 'Terminated', 0.00);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "3" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
INSERT INTO BANK_ACCOUNT (ACCOUNT_ID, EMPLOYEE_ID, PAYEE_ID,
ACCOUNT_TYPE, INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO)
VALUES
(13, 1, NULL, EMPLOYEE, 2231, 333, 11112);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "4" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
INSERT INTO BANK_ACCOUNT (ACCOUNT_ID, EMPLOYEE_ID,PAYEE_ID,
ACCOUNT_TYPE, INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO)
VALUES
(14, 2, NULL, EMPLOYEE, 3787, 123, 22223);
exit;
EOF
 read -n 1 -s
 elif [ "$CHOICE" == "5" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
INSERT INTO JOB_TABLE (JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC,
BASE_PAY)
VALUES
(1, 1, 'Manager', 'Manages team', 16.55);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "6" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
```

```
INSERT INTO JOB_TABLE (JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC,
BASE_PAY)
VALUES
(2, 2, 'Secretary', 'Office Management', 17.20);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "7" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

INSERT INTO SALARY (SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE,
ANNUAL, ANNUAL_BONUS)
VALUES
(1, 1, 1, 10000, 10000, 2000);
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "8" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

INSERT INTO SALARY (SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE,
ANNUAL, ANNUAL_BONUS)
VALUES
(2, 2, 2, 11000, 11000, 1000);
exit;
EOF
  read -n 1 -s
 elif  [ "$CHOICE" == "9" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

 INSERT INTO DEDUCTIBLES (DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID,
PAY_START, PAY_END, DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE,
DEDUCTIBLE_AMOUNT)
VALUES
(100, 1, 1,TO_DATE('2024-09-13', 'YYYY-MM-DD'), TO_DATE('2024-09-27',
'YYYY-MM-DD'), 0.1, 'Tax', 4789.00 );
```

```
exit;
EOF
  read -n 1 -s
  elif  [ "$CHOICE" == "10" ]
  then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

 INSERT INTO DEDUCTIBLES (DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID,
PAY_START, PAY_END, DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE,
DEDUCTIBLE_AMOUNT)
VALUES
(200, 2, 2, TO_DATE('2024-09-13', 'YYYY-MM-DD'), TO_DATE('2024-09-27',
'YYYY-MM-DD'), 0.2, 'Insurance', 5987.00 );
exit;
EOF
  read -n 1 -s
  elif  [ "$CHOICE" == "11" ]
  then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

 INSERT INTO DOCUMENT_TABLE (DOC_ID, EMPLOYEE_ID, DOC_TYPE,
DOC_NAME, ISSUE_DATE, EXPIRY_DATE) VALUES
(100, 1, 'Employment Contract', 'Contract_2024ver',
TO_DATE('2024-09-13','YYYY-MM-DD'), TO_DATE('2024-09-14', 'YYYY-MM-DD'));
exit;
EOF
  read -n 1 -s
  elif  [ "$CHOICE" == "12" ]
  then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

 INSERT INTO DOCUMENT_TABLE (DOC_ID, EMPLOYEE_ID, DOC_TYPE,
DOC_NAME, ISSUE_DATE, EXPIRY_DATE) VALUES
(200, 2, 'Identification', 'Driver’s License', TO_DATE('2024-09-12','YYYY-MM-DD'),
TO_DATE('2024-09-13', 'YYYY-MM-DD'));
exit;
EOF
  read -n 1 -s
  elif  [ "$CHOICE" == "13" ]
  then
```

```
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

 INSERT INTO PAYROLL (PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID,
APPLICABLE_DATE, PAYROLL_START, PAYROLL_END, PAYCODE,
HOURS_WORKED, MULTIPLIER)
VALUES
(1, 1, 1, TO_DATE('2024-10-04', 'YYYY-MM-DD'), TO_DATE('2024-09-13',
'YYYY-MM-DD'), TO_DATE('2024-09-27', 'YYYY-MM-DD'), 'regular', 80, 1.0);
exit;
EOF
  read -n 1 -s
  elif  [ "$CHOICE" == "14" ]
  then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca )(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

 INSERT INTO PAYROLL (PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID,
APPLICABLE_DATE, PAYROLL_START, PAYROLL_END, PAYCODE,
HOURS_WORKED, MULTIPLIER)
VALUES
(2, 2, 2, TO_DATE('2024-10-04', 'YYYY-MM-DD'), TO_DATE('2024-09-13',
'YYYY-MM-DD'), TO_DATE('2024-09-27', 'YYYY-MM-DD'), 'stat', 65, 2.0);
exit;
EOF
  read -n 1 -s

 elif [ "$CHOICE" == "E" ]
 then
 exit
 fi
 done
}


#--COMMENTS BLOCK--
ProgramStart()
{
 StartMessage
 while [ 1 ]
 do
 MainMenu

 done
```

```
}
# Start the program
ProgramStart
```

Queries.sh (4)

```
#!/bin/sh
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib

MainMenu()
{

 while [ "$CHOICE" != "START" ]
 do
 clear
 echo
"================================================================
="
 echo "| Oracle All Inclusive Tool|"
 echo "| Main Menu - Select Desired Operation(s):|"
 echo "| <CTRL-Z Anytime to Enter Interactive CMD Prompt>|"
 echo "-----------------------------------------------------------------"
 echo " Select Query:  "
 echo " $IS_SELECTED1 1) SELECT EMPLOYEE"
 echo " $IS_SELECTED2 2) SELECT BANK ACCOUNT"
 echo " $IS_SELECTED3 3) SELECT JOB TABLE"
 echo " $IS_SELECTED4 4) SELECT SALARY"
 echo " $IS_SELECTED5 5) SELECT DEDUCTIBLES"
 echo " $IS_SELECTED6 6) SELECT DEDUCTIBLES PAYEE"
 echo " $IS_SELECTED7 7) SELECT DOCUMENTS"
 echo " $IS_SELECTED8 8) SELECT PAYROLL"
 echo " $IS_SELECTED9 9) ADVANCED QUERIES"
 echo " $IS_SELECTEDa a) ALL"
 echo " $IS_SELECTEDX X) Force/Stop/Kill Oracle DB"
 echo " "
 echo " $IS_SELECTEDE E) End/Exit"
 echo "Choose: "
 read CHOICE
if [ "$CHOICE" == "0" ]
 then
 echo "Nothing Here"
exit;
```

```
        echo “Press any key to continue…”
        read -n 1 -s
 elif [ "$CHOICE" == "1" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
SELECT
    EMPLOYEE_ID, FIRST_NAME, LAST_NAME, DATE_OF_BIRTH, SIN_NUM,
ADDRESS, EMPLOYMENT_STATUS, PERFORMANCE_BONUS
FROM
    EMPLOYEE
WHERE
    EMPLOYEE_ID = 1;
exit;
EOF
read -n 1 -s

 elif [ "$CHOICE" == "2" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
SELECT
    ACCOUNT_ID, EMPLOYEE_ID,PAYEE_ID, ACCOUNT_TYPE,
INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO
FROM
    BANK_ACCOUNT
WHERE
    ACCOUNT_ID = 1 AND ACCOUNT_NO = 11111;
exit;
EOF
 read -n 1 -s
 elif [ "$CHOICE" == "3" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
SELECT
    JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC, BASE_PAY
FROM
    JOB_TABLE
WHERE
    EMPLOYEE_ID = 1 AND JOB_ID = 1;
exit;
EOF
```

```
read -n 1 -s
 elif [ "$CHOICE" == "4" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

SELECT
   SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE, ANNUAL,
ANNUAL_BONUS
FROM
   SALARY
WHERE
   HOURLY_RATE= 10000 AND SALARY_ID = 1;
exit;
EOF
read -n 1 -s
 elif [ "$CHOICE" == "5" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

SELECT
   DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID, PAY_START, PAY_END,
DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE, DEDUCTIBLE_AMOUNT
FROM
   DEDUCTIBLES
WHERE
   DEDUCTIBLES_ID = 1;
 exit;
EOF
read -n 1 -s
 elif [ "$CHOICE" == "6" ]
 then
sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

SELECT
   PAYEE_ID, DEDUCTIBLES_ID, PAYEE_NAME
FROM
   DEDUCTIBLES_PAYEE
WHERE
   PAYEE_ID = 101 AND DEDUCTIBLES_ID = 1;
 exit;
```

```
EOF
read -n 1 -s
 elif [ "$CHOICE" == "7" ]
 then
 sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
SELECT
    DOC_ID, EMPLOYEE_ID, DOC_TYPE, DOC_NAME, ISSUE_DATE,
EXPIRY_DATE
FROM
    DOCUMENT_TABLE
WHERE
    EMPLOYEE_ID = 1;
exit;
EOF
  read -n 1 -s
 elif [ "$CHOICE" == "8" ]
 then
 sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
SELECT
    PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID, APPLICABLE_DATE,
PAYROLL_START, PAYROLL_END, PAYCODE, HOURS_WORKED, MULTIPLIER
FROM
    PAYROLL
WHERE
    PAYROLL_ID = 1 AND HOURS_WORKED > 0;
exit;
EOF
read -n 1 -s
elif [ "$CHOICE" == "9" ]
 then
 sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
(SELECT
        EMPLOYEE.EMPLOYEE_ID,
        EMPLOYEE.FIRST_NAME,
        EMPLOYEE.LAST_NAME,
       'Employee' AS SOURCE_TYPE FROM EMPLOYEE)
UNION
(SELECT
       DOCUMENT_TABLE.EMPLOYEE_ID,
       DOCUMENT_TABLE.DOC_NAME AS FIRST_NAME,
      DOCUMENT_TABLE.DOC_TYPE AS LAST_NAME,
     'Document' AS SOURCE_TYPE
FROM
     DOCUMENT_TABLE);

(SELECT JOB_ID, JOB_NAME, JOB_DESC
FROM JOB_TABLE)

MINUS

(SELECT JOB_ID, JOB_NAME, JOB_DESC
FROM EMPLOYEE
JOIN JOB_TABLE ON EMPLOYEE.EMPLOYEE_ID = JOB_TABLE.EMPLOYEE_ID);

SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME
FROM EMPLOYEE E
WHERE EXISTS (
            SELECT 1
            FROM SALARY S
            WHERE S.EMPLOYEE_ID = E.EMPLOYEE_ID
            AND S.ANNUAL > 45000);
exit;
EOF
read -n 1 -s
 elif [ "$CHOICE" == "a" ]
 then
 sqlplus64
"Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=orac
le.cs.torontomu.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF
SELECT
   EMPLOYEE_ID, FIRST_NAME, LAST_NAME, DATE_OF_BIRTH, SIN_NUM,
ADDRESS, EMPLOYMENT_STATUS, PERFORMANCE_BONUS
FROM
   EMPLOYEE
WHERE
   EMPLOYEE_ID = 1;

SELECT
   ACCOUNT_ID, EMPLOYEE_ID,PAYEE_ID, ACCOUNT_TYPE,
INSTITUTION_NO, TRANSIT_NO, ACCOUNT_NO
FROM
   BANK_ACCOUNT
WHERE
   ACCOUNT_ID = 1 AND ACCOUNT_NO = 11111;

SELECT
```

```
   JOB_ID, EMPLOYEE_ID, JOB_NAME, JOB_DESC, BASE_PAY
FROM
   JOB_TABLE
WHERE
   EMPLOYEE_ID = 1 AND JOB_ID = 1;

SELECT
   SALARY_ID, EMPLOYEE_ID, JOB_ID, HOURLY_RATE, ANNUAL,
ANNUAL_BONUS
FROM
   SALARY
WHERE
   HOURLY_RATE= 10000 AND SALARY_ID = 1;

SELECT
   DEDUCTIBLES_ID, EMPLOYEE_ID, SALARY_ID, PAY_START, PAY_END,
DEDUCTIBLE_RATE, DEDUCTIBLE_TYPE, DEDUCTIBLE_AMOUNT
FROM
   DEDUCTIBLES
WHERE
   DEDUCTIBLES_ID = 1;

SELECT
   PAYEE_ID, DEDUCTIBLES_ID, PAYEE_NAME
FROM
   DEDUCTIBLES_PAYEE
WHERE
   PAYEE_ID = 101 AND DEDUCTIBLES_ID = 1;

SELECT
   DOC_ID, EMPLOYEE_ID, DOC_TYPE, DOC_NAME, ISSUE_DATE,
EXPIRY_DATE
FROM
   DOCUMENT_TABLE
WHERE
   EMPLOYEE_ID = 1;

SELECT
   PAYROLL_ID, SALARY_ID, DEDUCTIBLES_ID, APPLICABLE_DATE,
PAYROLL_START, PAYROLL_END, PAYCODE, HOURS_WORKED, MULTIPLIER
FROM
   PAYROLL
WHERE
   PAYROLL_ID = 1 AND HOURS_WORKED > 0;

(SELECT
        EMPLOYEE.EMPLOYEE_ID,
        EMPLOYEE.FIRST_NAME,
        EMPLOYEE.LAST_NAME,
       'Employee' AS SOURCE_TYPE FROM EMPLOYEE)
UNION
(SELECT
       DOCUMENT_TABLE.EMPLOYEE_ID,
       DOCUMENT_TABLE.DOC_NAME AS FIRST_NAME,
       DOCUMENT_TABLE.DOC_TYPE AS LAST_NAME,
      'Document' AS SOURCE_TYPE
FROM
      DOCUMENT_TABLE);

(SELECT JOB_ID, JOB_NAME, JOB_DESC
FROM JOB_TABLE)

MINUS

(SELECT JOB_ID, JOB_NAME, JOB_DESC
FROM EMPLOYEE
JOIN JOB_TABLE ON EMPLOYEE.EMPLOYEE_ID = JOB_TABLE.EMPLOYEE_ID);

SELECT
    dp.payee_name,
    SUM(d.deductible_amount) AS total_deductible_amount
FROM
         deductibles_payee dp
    JOIN deductibles d ON dp.deductibles_id = d.deductibles_id
GROUP BY
    Dp.payee_name;

SELECT
    e.employee_id,
    e.first_name,
    e.last_name,
    SUM(d.deductible_amount) AS insurance_amount
FROM
         employee e
    JOIN deductibles d ON e.employee_id = d.employee_id
WHERE
    d.deductible_type = 'Insurance'
GROUP BY
    e.employee_id,
    e.first_name,
    e.last_name
HAVING
```

```
    SUM(d.deductible_amount) > 100;
exit;
EOF
 elif [ "$CHOICE" == "E" ]
 then
 exit
 fi
 done
}

#--COMMENTS BLOCK--
ProgramStart()
{
 StartMessage
 while [ 1 ]
 do
 MainMenu
 done
}
#Start the program
ProgramStart
```