IF

NOV 2013

FACULTAD: INGENIERIA INDUSTRIAL Y DE SISTEMAS

INFORME FINAL DEL PROYECTO DE INVESTIGACIÓN

"ESTUDIO COMPARATIVO DE LOS LENGUAJES DE PROGRAMACIÓN ALGEBRAICOS SQL 2005 Y ORACLE."

INVESTIGADOR RESPONSABLE: GARCIA DIAZ, BERTILA LIDUVINA

PERIODO DE EJECUCIÓN: 01 DE OCTUBRE DEL 2011 AL 30 DE SETIEMBRE DEL 2013

RESOLUCIÓN RECTORAL DE APROBACION DEL PROYECTO Nº 1070-2011-R.- CALLAO DEL 02 DE NOVIEMBRE DEL 2011

CALLAO - PERU

I.- INDICE

II.- RESUMEN

"Estudio comparativo de los lenguajes de programación algebraicos SQL 2005 y ORACLE".

El objetivo de esta investigación es realizar un estudio comparativo a nivel práctico de ambos lenguajes algebraicos y profundizar en un tema que corresponde al curso de Base de Datos a mi cargo. Para recopilar los datos de este estudio se creó una Base de Datos en cada uno de los Lenguajes y se comprobó a nivel del Lenguaje de definición de datos (DDL), Lenguaje de manipulación de datos (DML) y Lenguaje de control de datos (DCL) sus similitudes y diferencias.

De los resultados obtenidos en la investigación podemos concluir que las diferencias entre ambos lenguajes es mayor a nivel de Lenguaje de definición de datos (DDL), siendo mínima la diferencia en el Lenguaje de Manipulación de Datos (DML) y existen pequeñas diferencias a nivel del Lenguaje de Control de Datos (DCL).

Existen diferencias a nivel de los sistemas operativos donde corren estos SGBD, mientras que ORACLE es multiplataforma, SQL server sólo se ejecuta en el Sistema Operativo de Windows de Microsoft.

Existen diferencias en cuanto a costos, ya que el costo de utilizar el ORACLE es mayor que en SQL server.

Desde el punto de vista del software, ORACLE ofrece más ventajas, como por ejemplo el uso de DATAWAREHOUSE y MINERIA DE DATOS, que forman parte del Lenguaje. SQL también busca mejorar estas características. Ambos utilizan el Modelo Relacional.

Las conclusiones obtenidas pueden ser utilizadas por cualquier investigador de dicho tema o de alguna rama de la ingeniería del software.

III.- INTRODUCCIÓN

3.1 PLANTEAMIENTO DEL PROBLEMA

"Estudio comparativo de los lenguajes de programación algebraicos SQL 2005 y ORACLE".

Esta investigación es una investigación de tipo **cualitativo**, donde se compara estos 2 lenguajes de programación algebraicos como es SQL 2005 y ORACLE con el objetivo de comprobar sus similitudes y diferencias a nivel de software, con el fin de poder decidir cuál utilizar dado un contexto determinado o partir de uno de ellos, para conocer el otro.

Considero este tema muy importante por ser estos Lenguajes algebraicos los responsables de la mejora en la gestión de las empresas y cuyo manejo es vital para el desarrollo laboral de los futuros ingenieros de sistemas.

Ambos son los Gestores de Base de Datos más utilizados para gestionar las Empresas y uno de los principales motivos de la existencia de la Ingeniería de Sistemas, ya que esta sin herramientas de trabajo como los lenguajes de programación algebraicos, tendría como herramientas sólo un editor como Word o una hoja de cálculo como Excel, y no hubiera alcanzado el desarrollo científico y tecnológico que hoy tiene.

Sin embargo, es importante analizarlos a través de aplicaciones prácticas y poder inferir nuestras propias conclusiones acerca de las similitudes y diferencias, de las ventajas y desventajas de ambos lenguajes algebraicos.

En resumen el planteamiento del problema sería el siguiente:

a.- ¿Qué analogías y que diferencias existen entre el Lenguaje Algebraico SQL 2005 y ORACLE?

b.- ¿Cuál Lenguaje Algebraico convendría más impartir en el Curso de Bases de Datos en los momentos actuales?.

c.- ¿Se podría usar SQL 2005 en el curso de Base de Datos y ORACLE en un curso de Taller de Base de Datos o usar ambos en el curso Base de Datos de acuerdo a un orden de dificultad teniendo en cuenta los cambios en las tecnologías, así como la utilidad comercial de ambos?

d.- ¿Cuáles son las diferencias en cuanto a requerimientos de Plataformas de Sistemas Operativos y hardware de SQL SERVER y ORACLE.

e.- ¿Qué ventajas tienen cada uno de estos Gestores de Bases de datos?

f.- ¿Qué desventajas tienen cada uno de estos Gestores de Bases de datos?

3.2 OBJETIVOS Y ALCANCES DE LA INVESTIGACIÓN

El objetivo principal es definir y determinar la importancia de los lenguajes algebraicos a través de un estudio comparativo. Correspondería a un tipo de investigación básica.

a. OBJETIVOS GENERALES

- Determinar la importancia de los Lenguajes Algebraicos, mediante un estudio comparativo
- Apoyar el dictado del curso de Base de Datos a mi cargo, en la Universidad Nacional del Callao.

b. OBJETIVOS ESPECÍFICOS

De acuerdo a las definiciones anteriores, esta investigación permitirá:

- Profundizar en la investigación de los diferentes Lenguajes de Programación Algebraicos.
- Investigar las diferencias y semejanzas que existen entre los diferentes Lenguajes de Programación Algebraicos.
- Evaluar la utilidad y beneficio de cada uno de los Lenguajes de Programación Algebraicos.

c. ALCANCES DE LA INVESTIGACIÓN

El tipo de Investigación será realizar el curso de Base de Datos a mi cargo, utilizando estos Lenguajes de Programación Algebraicos, para poder analizar sus efectos en la Gestión de Bases de Datos.

El sector que se verá beneficiado con los resultados de esta investigación son: los alumnos de la Facultad de Ingeniería Industrial y de Sistemas de la UNAC.

3.3 IMPORTANCIA Y JUSTIFICACION

Este proyecto es importante porque no existe un trabajo similar que le preceda. Una vez que ha sido demostrado los resultados de esta relación tendrán amplia generalización.

Para ejecutar la presente investigación, se hará uso de las teorías científicas que a continuación se indican: Modelo Conceptual, Modelo Relacional, SQL.

El objetivo principal es definir y determinar la importancia de los lenguajes algebraicos a través de un estudio comparativo. Correspondería a un tipo de investigación básica. Este proyecto por corresponder a las ciencias básicas, requiere de justificación teórica.

Es la motivación para emprender el siguiente trabajo de investigación, estrechamente ligado al curso de Base de Datos que que tengo a mi cargo en la Escuela de Ingeniería de Sistemas de la Universidad Nacional del Callao.

IV.- MARCO TEORICO

En cuanto a las teorías y/o leyes y/o doctrinas estrechamente relacionadas con el problema y objetivos dados en el proyecto de investigación, de acuerdo a la operacionalización de la hipótesis del proyecto de investigación, existen las siguientes variables independientes: Lenguajes algebraicos, SQL, SQL 2005, ORACLE que son necesarias conocer para demostrar la variable dependiente que es el estudio comparativo de ambos Lenguajes Algebraicos, las cuales expondré a continuación.

Existen trabajos anteriores que tengan el mismo objetivo que el presente trabajo de investigación a nivel teórico, las cuales expondré a continuación, pero **no existen trabajos previos a nivel práctico**, de repetir lo hechos y extraer sus propias conclusiones en el tema de Lenguajes algebraicos.

4.1 TEORIAS RELACIONADAS CON EL PROBLEMA Y OBJETIVOS DADOS EN EL PROYECTO DE INVESTIGACIÓN.

4.1.1 LENGUAJES DE PROGRAMACIÓN ALGEBRAICOS

Los Lenguajes Algebraicos son un tipo de lenguaje Declarativo, en la programación declarativa las sentencias que se utilizan lo que hacen es describir el problema que se quiere solucionar, pero no las instrucciones necesarias para solucionarlo. Esto último se realizará mediante mecanismos internos de inferencia de información a partir de la descripción realizada.

Los lenguajes declarativos están basados en la definición de funciones o relaciones. No utilizan instrucciones de asignación (sus variables no almacenan valores). Son los más fáciles de utilizar (no se requieren conocimientos específicos de informática), están muy próximos al hombre. Se suelen denominar también lenguajes de órdenes, ya que los programas están formados por sentencias que ordenan "qué es lo que se quiere hacer", no teniendo el programador que indicar a la computadora el proceso detallado (el algoritmo) de cómo hacerlo".

SQL es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas. Una de sus

características es el manejo del álgebra y el cálculo relacional que permiten efectuar consultas con el fin de recuperar de forma sencilla información de interés de bases de datos, así como hacer cambios en ella.

En 1976, Peter Chen presentó el modelo entidad-relación, que es la técnica más utilizada en el diseño de bases de datos. Estas teorías se complementan con las investigaciones del Dr. Codd, y sólo hay un paso para transformar este Modelo Entidad Relación en un Modelo Relacional y luego proceder a la creación de Diseño Físico con la ayuda de los mensionados lenguajes algebraicos.

4.1.1.1 MODELAMIENTO ENTIDAD – RELACIÓN

El modelo entidad relación (ER) proporciona una herramienta para representar información del mundo real a nivel conceptual, y el el modelo más utilizado. Creado en 1976 por Peter Chen, permite describir las entidades involucradas en una base de datos, así como las relaciones y restricciones de ellas. Chen presenta el modelo como una vista unificada de los datos, centrándose en la estructura lógica y abstracta de los datos, como representación del mundo real, con independencia de consideraciones de tipo físico.

"Este modelo y sus variaciones se emplean a menudo en el diseño conceptual de aplicaciones de bases de datos, y muchas herramientas de diseño de bases de datos aplican sus conceptos" (ELMASRY /NAVATHE, 2007).

4.1.1.2 MODELO RELACIONAL

En junio de 1970 el Dr. Codd, quien Trabajaba para IBM, publicó un paper con el título "un modelo relacional de datos para grandes bancos de datos compartidos". Uno de los objetivos centrales de este artículo se expresó así:

"*Los futuros usuarios de los grandes bancos de datos deben ser protegidos de tener que saber cómo se organizan los datos en la máqina (la representación interna)*" (REINOSA)(2012) .

Lo que el Dr. Codd deseaba era alejar a los usuarios del nivel físico, elevar el nivel de abstracción con el que debían interactuar los usuarios y

que no dependieran de los detalles técnicos para usar los datos almacenados.

De otro lado su amigo el reconocido Cristopher Date, especializado en la tecnología de bases de datos relacionales, hace un tributo después de la muerte del Dr. Codd diciendo:

"*El Modelo Relacional es ampliamente reconocido como una de las grandes innovaciones del siglo XX*" (REINOSA)(2012).

Y continua: *"El modelo relacional proporcionó un marco teórico en el que una variedad de problemas importantes podrían ser atacados de una manera científica"* (REINOSA)(2012).

El modelo relacional se ha establecido actualmente como el principal modelo de datos para las aplicaciones de procesamiento de datos. Ha conseguido la posición principal debido a su simplicidad, que facilita el trabajo del programador en comparación con otros modelos anteriores como el de red y el jerárquico.

Se basa en que la representación física deberá satisfacer y representar, de alguna forma, las relaciones y restricciones lógicas del esquema relacional.

Presenta la Base de Datos como una colección de relaciones, cuyo contenido varía en el tiempo.

Una base de datos relacional consiste en un conjunto de **tablas**, a cada una de las cuales se le asigna un nombre exclusivo. Cada fila de la tabla representa una ***relación*** entre un conjunto de valores.

Dado que cada tabla es un conjunto de dichas relaciones, hay una fuerte correspondencia entre el concepto de *tabla* y el concepto matemático de *relación*, del que toma su nombre el modelo de datos relacional.

Una relación puede ser vista como una tabla, con filas llamadas tuplas y con cabecera de columnas llamadas atributos.

"*Matemáticamente, una relación deinida sobre los n domiios* D_1, D_2, ...,D_n *no necesariamente distintos, es un subconjunto del producto cartesiano de estos dominios, donde cada elemento de la relación (tupla) es una serie de n valores ordenados*" (DE MIGUEL, Adoración y PIATTINI, Mario, 1999).

Con respecto a la parte dinámica del modelo, se propone un conjunto de operadores que se aplican a las relaciones (Algebra relacional y cálculo relacional).

Lo que realmente marca la diferencia entre los sistemas relacionales y los sistemas anteriores es el hecho de que su creador, Codd, basó expresamente su funcionamiento sobre un modelo matemático muy

específico: el álgebra relacional y el cálculo relacional, así como la progresiva adopción, por parte de su creador y algunos colaboradores, de un número de Reglas de Integridad Relacional y de Formas Normales.
A partir de 1980, el modelo relacional ha tenido un auge espectacular, gracias al desarrollo tecnológico, han ido apareciendo productos comerciales que corren en las más diversas plataformas con rendimientos aceptables: ORACLE, SQL, DB2, SYBASE.

1.- Propiedades Del Modelo Relacional

Entre las propiedades del Modelo Relacional tenemos:

- No existen tuplas repetidas.
- Corolario: Siempre existe una clave primaria
- Las tuplas no están ordenadas de arriba hacia abajo.
- Los atributos no están ordenados de izquierda a derecha.
- Todos los valores de los atributos son atómicos.
- Los atributos multivaluados se deben representar con relaciones individuales.

2.- Aspectos del Modelo Relacional

El Modelo Relacional se le puede estudiar a través de 3 aspectos:

- Estructura
- Integridad
- Manipulación

Por su Estructura

Por su estructura se define al Dominio como la base a partir de la cuál se construyen las relaciones, compuestas a su vez de una cabecera (esquema) y un cuerpo (ó extensión) de tuplas, así como de una clave primaria.
Un *esquema* de relación R, denotado por $R(A_1,A_2,..A_n)$ se compone de un nombre de relación R y una lista de atributos $A_1,A_2,..A_n$.
Ejemplo: ESTUDIANTE(Nombre, NSS, TelPârticular, Dirección.Edad, Prom).
Los dominios son estáticos, las relaciones dinámicas (el contenido cambia en el tiempo)
Dada una serie de conjuntos $D_1,D_2,...D_n$, se dice que R es una relación entre estos n conjuntos si es un conjunto de n tuplas no ordenadas

$(d_1,d_2,..d_n)$ tales que $d_1 \in D_1$, $d_2 \in D_2$,...$d_n \in D_n$. A los conjuntos $D_1,D_2,...D_N$ se les denomina dominios de R y el valor n es el grado de la relación R

Conceptos y equivalencias:

Relación:	tabla
Tupla:	Fila de la Tabla
Cardinalidad:	# de tuplas de una relación
Grado:	# de atributos de una relación
Dominio:	Colección de valores de un atributo
Atributo:	Columna de la tabla

Ejemplo:

codAg	activo	ciudad
1	500,000	Lima
2	300,000	Ica

Relación: AGENCIA Cardinalidad = 2 Grado = 3

Dominio de ciudad = Lima, Ica, Arequipa, Tacna....

Atributo = codAg, activo, ciudad

Por su Integridad

Se la puede ver como las propiedades que se encuentran en las definiciones de:

- Reglas de Entidad: unicidad, minimalidad
- Reglas de Integridad de Entidades
- Reglas de Integridad Referencial

Reglas De Entidad: Unicidad, Minimalidad

La Unicidad y la Minimalidad son 2 principios básicos asociados a la integridad de entidad.

La unicidad: En cualquier momento no existen dos tuplas en R (relación) con el mismo valor de clave K.

La minimalidad: Si la clave K es compuesta, no será posible eliminar ningún componente de K sin destruir la propiedad de unicidad.

Debe haber la menor cantidad de campos en la clave primaria.

Regla De Integridad De Entidades

Ningún componente de la clave primaria (PK) de una relación base puede aceptar nulos (NULL) que significa información faltante, desconocida.

Regla De Integridad Referencial

La clave foránea (FK) es un atributo de una relación R2, cuyos valores deben concordar con los de la clave primaria de alguna relación R1, con la cual existe un vínculo.

Ejemplo:

Agencia	Depósito
CodAg(PK)	CodAg(FK)
01	05
02	
03	

Con el término "*valores ajenos sin concordancia*" queremos decir aquí un valor no nulo de clave ajena para el cual no existe un valor concordante de la clave primaria en la relación objetivo pertinente.

Las Regla de integridad referencial garantizan que la base de datos no incluya valores no válidos de una clave foránea.

R2(referencial) → R1(Relación referida)

Relación Referencial.- es la relación que contiene la clave foránea (FK).

Relación Referida.- es la relación que contiene la clave primaria (PK).

Las claves foránea y primaria deben definirse sobre el mismo dominio.

Si B hace referencia a A entonces A debe existir.

Por su Manipulación

Se le puede ver como las operaciones que se ejecutan: El Algebra Relacional y el Cálculo Relacional.

4.1.2 SQL

Los orígenes del SQL están ligados a los de las bases de datos relacionales. En 1970 E. F. Codd propone el modelo relacional. Sin embargo, fue Oracle quien lo introdujo por primera vez en 1979 en un programa comercial. Hubo pues una demora entre la creación del modelo Relacional y su implementación computacional de aproximadamente 10 años, pues no había un software que soportará el Modelo. Oracle SQL fue un gran éxito y dio lugar a toda una industria en torno a SQL. Sybase, Informix, Microsoft y otras compañías se presentaron con sus implementaciones de un sistema de gestión de base de datos relacional basado en SQL (RDBMS).

El SQL pasa a ser el lenguaje por excelencia de los diversos sistemas de gestión de bases de datos relacionales surgidos en los años siguientes y es por fin estandarizado en 1986 por el ANSI, dando lugar a la primera versión estándar de este lenguaje. Al año siguiente este estándar es también adoptado por la ISO.

Debido a que el objetivo principal de SQL es comunicar las acciones en el servidor de base de datos, no tiene la flexibilidad de un lenguaje de propósito general. La mayoría de la funcionalidad de SQL son preocupaciones entrada y salida de la base de datos: añadir, modificar, eliminar y leer datos.

En la actualidad el SQL es el estándar *de facto* de la inmensa mayoría de los SGBD comerciales.

El lenguaje SQL cumple con las reglas del Modelo Relacional, establecidas por el Dr. Codd, que son usados por el ORACLE y el SQL 2005, que además tienen como antecedentes el Modelo Entidad Relación de Peter Chen.

El lenguaje SQL se puede considerar como una de las principales razones del éxito comercial de las bases de datos relacionales. Como se convirtió en un estándar para estas últimas, los usuarios perdieron el temor a migrar sus aplicaciones de base de datos desde otros tipos de sistemas de bases de datos a los sistemas relacionales.

El lenguaje SQL proporciona una interfaz de lenguaje declarativo del más alto nivel, por lo que el usuario sólo especifica lo que debe ser el resultado, dejando para el DBMS la optimización y las decisiones de cómo ejecutar la consulta.

El SQL tiene un rol muy importante en la comunicación con la base de datos, que también funciona embebido en otros lenguajes de uso general, como los lenguajes procedimentales: Cobol, C, C++ y los lenguajes orientados a objetos como: Java, PHP y Python.

SQL es un lenguaje de consultas estructurado que opera sobre bases de datos relacionales y la normalización.

"*Aunque el lenguaje SQL se considere un lenguaje de consultas, contiene muchas otras capacidades además de la consulta en bases de datos. Incluye características para definir la estructura de los datos, para la modificación de los datos en la base de datos y para la especificación de restricciones de seguridad.*" (SILBERSCHATZ, Abraham y KORTH, Henry, 2002).

SQL utiliza los términos *tabla, fila y columna* para los términos: *relación, tupla y atributo* del Modelo Relacional, respectivamente.
El principal comando de SQL para definir datos es la sentencia *create*, que se utiliza para crear esquemas, tablas y dominios, así como otras estructuras, como vistas, aserciones y triggers.
SQL es un lenguaje de bases de datos global: cuenta con sentencias para definir datos, consulta y actualizaciones. Se comporta como DDL (data definition language) y como DML (data manipulation language) y dispone de características para especificar temas de seguridad y autorización, definir restricciones de integridad y especificar controles de transacciones como el DCL (data control Language).

4.1.2.1 LENGUAJE DE DEFINICIÓN DE DATOS (DDL)

En el caso específico del DDL, el lenguaje del SGBD debe ser capaz de definir la estructura lógica de la Base de Datos, sin entrar en detalles de implementación ni mecanismos en que se accede a los datos de la Base de Datos.
La forma idónea de realizar lo anterior es mediante un lenguaje declarativo, el cual permite declarar la estructura del modelo de acuerdo al modelo de datos que utiliza el SGBD.

4.1.2.2 LENGUAJE DE MANIPULACION DE DATOS (DML)

Este lenguaje permite la utilización de los datos almacenados en la base de datos, y es usado generalmente por usuarios finales o por aplicaciones cliente utilizadas por ellos.
La forma más común de utilización del DML es mediante el uso de lenguajes de programación convencionales que poseen al DML del SGBD como lenguaje embedido. Así, se puede programar la aplicación cliente en el lenguaje que se elija, pudiendo realizar llamadas a sentencias del DML cuando sea necesario.
En el caso de los SGBDR y el lenguaje SQL, se tienen las siguientes sentencias de manipulación de datos:
Inserción de Datos (INSERT);
Eliminación de Datos (DELETE);
Modificación de Datos (UPDATE);

Consulta de Datos (SELECT).

El lenguaje del SGBD debe incluir formas de especificar qué se desea hacer con los datos (insertar, recuperar, modificar o borrar datos), sin entrar en detalles acerca de cómo se realizan estas operaciones.

4.1.2.3 LENGUAJE DE CONTROL DE DATOS (DCL)

Son sentencias que permiten controlar el desarrollo de una transacción con el SGBD, de manera que el usuario pueda, a medida que se va realizando la transacción, confirmarla o retractarse de ella.

En el caso de los SGBDR y el lenguaje SQL, se tienen las siguientes sentencias de control de transacciones:

Confirmación de transacción (COMMIT);

Retractación de Transacción (ROLLBACK);

Almacenamiento de puntos intermedios de transacciones (SAVEPOINT).

4.1.3 SQL 2005

Tenemos diferentes versiones de SQL Server 2005, cada una orientada a cubrir unas determinadas necesidades de diferentes tipos de empresas o clientes, SQL server es uno de las Bases de datos Relacionales más usadas actualmente por su menor costo y la facilidad para su aprendizaje.

4.1.3.1 ESTRUCTURACIÓN DE LA BASE DE DATOS

Estructura Física

En empresas cuyo volumen de datos es altísimo y el trabajo que se realiza sobre la base de datos soporta una actividad elevada, se almacenan los archivos en grupos de discos denominados RAID por hardware. Este método mejora considerablemente el rendimiento, y nos asegura que en caso de fallos inesperados no perdamos información.

SQL 2005 tiene dos archivos donde almacenar la base de datos:

- Archivo de datos.
- Archivo de registro de transacciones.

Archivo de datos.- o aquellos que añadimos como extras, son los archivos que tendrán almacenada la información, los datos. SQL Server 2005 nos permite también crear en nuestras bases de datos, no sólo información,

sino también una serie de objetos que trabajan con la información. Pues bien, esta serie de objetos también se almacena en el archivo de datos.

Archivo de registro de transacciones.- Este fichero es tan importante como el anterior. Su importante tarea es garantizar que esa base de datos permanezca integra. Gracias a estos archivos de registros (puede haber más de uno), en caso de ser necesario, podremos recuperar la base de datos, ya que almacena las modificaciones que se producen debido a la actividad o la explotación de la base de datos.

4.1.3.2 TAMAÑO DE LA BASE DE DATOS

En SQL Server 2005. Los archivos de datos y de registro, crecen automáticamente. No crecen con cada dato que se añade. Nosotros como administradores, le daremos un tamaño inicial sencillo de estimar (una cantidad muy pequeña, unos Megabytes), en ese momento SQL Server 2005 crea la estructura correcta para la base de datos, y una vez que nuestra base de datos está en explotación cuando alcanza el tamaño limite, lo incrementa una cantidad dada por un factor predeterminado.

4.1.3.3 LENGUAJE DE DEFINICION DE DATOS (DDL)

SQL server se puede utilizar para realizar procesamiento de transacciones, almacenar y analizar datos, y generar nuevas aplicaciones de Bases de Datos.
Tiene como principales funciones: crear tablas, implementar constraints para controlar la integridad de datos.

Tipo de integridad	*tipo de constraint*
Controlar rango de valores sobre las columnas	DEFAULT
	CHECK
Unicidad de registros y valores únicos	PRIMARY KEY
	UNIQUE
Referencial	FOREIGN KEY
	CHECK

Figura 4.1 Tipos de constraints. Fuente: Autor

4.1.3.4 LENGUAJE DE MANIPULACION DE DATOS (DML)

El lenguaje DML permite la inserción, actualización, borrado y selección de datos y es una de las funciones más importantes en la gestión de una base de datos, sobretodo la sentencia select, ya que permite a los Gerentes realizar consultas rápidas acerca de sus negocios.

4.1.3.5 LENGUAJE DE CONTROL DE DATOS (DCL)

Las declaraciones del DCL se usan para cambiar los permisos o roles asociados con un usuario de la base de datos. Y son: **GRANT, DENY y REVOKE**

4.1.4 ORACLE

La última versión de Oracle es la versión 11g, liberada en el mes de julio de 2009, es un RDBMS portable ya que se puede instalar en los sistemas operativos más comunes en el mercado, el costo de la licencia oscila entre los 180 y 400 dólares dependiendo del tipo de licencia de usuario, soporta hasta 4 peta bytes de información. Cuenta con administración de usuarios así como la administración de roles, además soporta trigers y store procedure, cuenta con conectividad JDBC y ODBC, siempre y cuando se tengan los drivers adecuados para la misma. Es un DBMS seguro ya que cuenta con un proceso de sistema de respaldo y recuperación de información. Soporta Data Warehouse por lo que facilita el acceso a la información y da mayor versatilidad. La mayor parte de las empresas de telecomunicaciones en Latinoamérica utilizan Oracle, por lo que se puede decir que es un DBMS confiable, seguro para ser utilizado en una empresa y sobre todo permite reducir costos por su accesibilidad en el mercado.

Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando su:

- Soporte de transacciones.
- Estabilidad.
- Escalabilidad.
- Soporte multiplataforma.

La base de datos Oracle en Windows ha evolucionado desde un nivel básico de integración del sistema operativo hasta utilizar servicios más avanzados en la plataforma Windows.

4.1.4.1 CARACTERISTICAS DE ORACLE

El 3 de septiembre de 2007, Oracle anunció el nuevo sistema de administración de base de datos, Oracle Database 11g, en el mercado de Japón. De acuerdo con la visión de brindar calidad incomparable, facilidad de uso al usuario y eficacia en la administración de datos, Oracle Database 11g fue desarrollada con la máxima capacidad de recursos de ingeniería para un solo producto. Oracle Database 11g es el resultado de un proceso de desarrollo que incorpora las opiniones de grupos de usuarios y partners para brindar un mejor desempeño, seguridad y administración automatizada. Como base de datos de próxima generación, el producto presenta las siguientes características:

SecureFiles

Real Application Testing

Advanced Compression

Total Recall

SecureFiles.-

Permite que la gran cantidad de información de objetos no estructurados, como las imágenes de rayos X para atención médica y los gráficos de ingeniería que en el pasado eran ejecutados en el sistema de archivos, sean directamente almacenados y administrados dentro de la base de datos para resolver los cuellos de botella que se presentan durante el desempeño. Soporta las funciones transparentes de comprensión, seguridad y administración de base de datos. Al contar con toda la información empresarial, incluso los procedimientos y funciones, en una sola base de datos definida.

Real Application Testing.-

Es probablemente la opción más innovadora de la última edición empresarial de Oracle Database 11g. La función puede capturar cargas de trabajo de base de datos en tiempo real y repetirlas en un entorno de prueba. Esto ayuda a los usuarios a probar y personalizar sus sistemas para alcanzar un óptimo desempeño.

Esta función también es útil cuando el usuario reemplaza el hardware o el Sistema Operativo y desea probar el impacto en el sistema.

Total Recall.-

Permite a los usuarios rastrear y mantener los cambios de datos de manera fácil y económica, a través del archivo continuo de datos. En el ambiente mundial, donde el cumplimiento ha pasado a primera plana, la capacidad de recuperar datos rápidamente cuando se lo solicita se considera muy valiosa. En el pasado, era extremadamente costoso, aunque necesario, almacenar cantidades masivas de datos en sistemas de almacenamiento.

Advanced Compression.-

Es una tecnología versátil que comprime efectivamente tablas masivas y datos no estructurados. Reduce las capacidades de almacenamiento y obtiene una gran cantidad de otros beneficios como cargas reducidas de red y mejor eficacia de backup. La función permite a los usuarios ahorrar costos en sistemas, capacidad y espacio.

4.1.4.2 OTRAS CARACTERISTICAS DE ORACLE

1.- *Tablespace*

Detras de las construcciones logicas asociadas con una base de datos Oracle, existen construcciones fisicas que la base de datos Oracle utiliza para almacenar los datos.

Sin embargo, los tablespaces no solo constituyen contenedores de objetos, tambien son un metodo de organización. Los tablespaces estan diseñados para organizar los datos permitiendo al DBA agrupar tipos de datos similares en un área. Por ejemplo, un DBA puede utilizar un tablespace para almacenar datos y otro para almacenar indices.

2.- Sentencias SQL

A continuación presentamos un resumen de las sentencias SQL de ORACLE.

SELECT	Recuperación de Datos
INSERT → Insertar UPDATE → Actualizar DELETE → Borrar MERGE → Ordenar	DML (Manipulación de Datos)
CREATE → Crear ALTER → Modificar DROP → Borrar RENAME → Renombrar TRUNCATE → Eliminar datos	DDL (Data Definition control)
COMMIT → Actualizar cambios ROLLBACK → Deshacer cambios SAVEPOINT → punto de recuperación	Control de transacciones
GRANT → Otorgar permisos REVOKE → Revocar permisos	DCL(Data Control Language)

Figura 4.2 Sentencias SQL. Fuente: Autor

4.1.4.3 ARQUITECTURA DE ORACLE EN WINDOWS

Cuando se ejecuta en Windows, Oracle Database 11g presenta las mismas características y la misma funcionalidad que las distintas plataformas Linux y UNIX soportadas por Oracle.

No obstante, la interface entre la base de datos y el sistema operativo ha sido sustancialmente modificada para aprovechar los servicios exclusivos brindados por Windows. Como resultado, Oracle Database 11g en Windows no es un puerto directo de la base de código UNIX.

Oracle brinda buen desempeño para las plataformas Windows de 32 y 64 bits.

Oracle se ha adaptado a los cambios más recientes en los conocimientos informáticos de Windows desde las primeras soluciones cliente/servidor hasta las aplicaciones de Internet, y ahora la tecnología grid.

Oracle Database 11g para Windows brinda todas las características necesarias para la administración de datos, ya sea que se utilice para la implementación en toda la empresa o en un solo departamento. Permite a los usuarios aprovechar las ventajas de facilidad de uso y costo que ofrece Windows, mientras brinda la escalabilidad, confiabilidad y el desempeño. Uno de los mayores beneficios de utilizar Oracle es el soporte de múltiples

estándares de programación. Al soportar aplicaciones Java, .NET, PHP y C/C++, Oracle garantiza que todos los desarrolladores puedan utilizar las características avanzadas de la base de datos Oracle.

4.1.4.4 LENGUAJE DE DEFINICION DE DATOS (DDL)

Las sentencias DDL en ORACLE son:
CREATE TABLE ⟶ Crea una tabla. Para ello el usuario debe de tener el privilegio *CREATE TABLE*.
ALTER TABLE ⟶ Permite modificar la estructura definida para una tabla.
DROP TABLE ⟶ Elimina una tabla (datos y estructura) y sus índices. No se puede hacer Rollback de esta sentencia.
RENAME ⟶ Cambia el nombre de una tabla, vista, secuencia o sinónimo.

4.1.4.5 LENGUAJE DE MANIPULACION DE DATOS (DML)

Las declaraciones de lenguaje de manipulación de datos (DML) se utilizan para la gestión de datos dentro de los objetos de esquema, las usadas en Oracle son:

INSERT ⟶ Añade registros a una tabla.
UPDATE ⟶ Modifica registros existentes de una tabla.
DELETE ⟶ Elimina registros existentes de una tabla.

4.1.4.6 LENGUAJE DE CONTROL DE DATOS (DCL)

El *Data Control Language*, permite configurar controles de seguridad sobre objetos de la base de datos. Básicamente a este grupo de sentencias pertenecen los comandos GRANT y REVOKE, los que respectivamente otorgan y quitan privilegios de operaciones DDL o DML sobre ciertos objetos de la base de datos de algún esquema en particular. Este tipo de comandos no requieren confirmación del usuario (COMMIT).

4.2 TRABAJOS DE INVESTIGACIÓN TEORICOS ANTERIORES

Las organizaciones a menudo emplean un sin número de plataformas de base de datos en su arquitectura de sistemas de información. Utilizan de tres a cuatro diferentes paquetes de Base de Datos.

Sin embargo cuando se trata de aprender una plataforma nueva base de datos, a menudo es mejor mirar a lo que ya sabemos y luego tratar de encontrar el equivalente en el nuevo entorno. Ejemplo: si un desarrollador de SQL Server desea escribir procedimientos almacenados de Oracle puede empezar mirando las funciones integradas y en qué difieren.

Mostraremos algunas de las similitudes y diferencias entre las dos plataformas de base de datos más utilizadas hoy en día.

4.2.1 EDICIONES

En la actualidad SQL Server 2012 es la versión actual del producto de base de datos de Microsoft. La versión anterior fue SQL Server 2008 R2, que tuvo como su antecedente SQL Server 2005, la cual tuvo una importante actualización de su predecesor, SQL Server 2000.

La versión más utilizada sigue siendo SQL Server 2008 R2, tanto en Empresas como en Universidades.

Oracle por otro lado se encuentra en la versión 11g R2. La versión más prominente de 10g R2 ha estado presente en el mercado por algún tiempo.

En cuanto a ediciones de SQL Server 2008 R2 ofrece las siguientes:

- **Enterprise Edition**: La edición Enterprise es recomendable para base de datos de alto volumen.
- **Standard Edition**: Es ideal para las empresas que no requieren de las funciones avanzadas de la Enterprise Edition. Lo usan la mayoría de las empresas.
- **Workgroup Edition**: Es la indicada para pequeñas aplicaciones departamentales.
- **Web Edition**: Esta destinada a ser utilizada por los proveedores de alojamiento como solución back-end de bajo coste para aplicaciones web.

- **Express Edition**: Puede ser utilizados para el almacenamiento local de datos y sistema de desarrollo de pequeñas escala. La edición Express se puede descargar gratis.
- **Compact Edition**: La edición Compact permite a los usuarios desarrollar aplicaciones para computadoras de escritorio de Windows y dispositivos de mano.
- **Developer Edition**: Tiene licencia de uso por un usuario a la vez y se utiliza para fines de desarrollo y pruebas.

Para Oracle 11g R2, las Ediciones son:

- **Enterprise Edition**: Al igual que en el SQL Server Enterprise Edition, todas las características y las capacidades del producto están permitido en esta edición.
- **Standard Edition**: Al igual que el estándar SQL Server Edition, la edición estándar de Oracle tiene habilitadas las principales características del producto.
- **Standard Edition One**: Esta edición está diseñada para pequeños grupos de trabajo y la licencia para un máximo de 5 usuarios.
- **Express Edition**: Trabaja en pequeña escala, también tiene licencia para redistribuirlo libremente.

4.2.2 SISTEMAS OPERATIVOS

Microsoft SQL Server ha tenido como plataforma los servidores de Windows. En la actualidad, SQL Server hasta la versión 8 se ejecutan en XP, Vista, Windows Server 2000, 2003 y 2008.

A partir de la versión 12 se ejecutan solo en los sistemas operativos: Windows 7, Windows 8.

La plataforma de base de datos está disponible para los de 32 bits y 64 bits de Windows.

Para el caso de Oracle, tiene soporte multiplataforma que incluye no sólo Windows (32 bits y 64 bits), también Linux y diferentes variantes de Unix (Solaris, HP-UX, AIX, etc).

4.2.3 ARQUITECTURA

4.2.3.1 INSTANCIAS Y BASES DE DATOS

Una instancia en términos de SQL Server, significa un servicio de aplicación autocontenida que implica archivos del sistema operativo, las estructuras de memoria, los procesos de segundo plano y la información de registro. Una instancia está representada por un servicio en Windows y puede estar en ejecución o estado detenido. Cuando se ejecuta, una instancia ocupa una porción de la memoria del servidor.

Una base de datos de SQL Server es el repositorio de datos y el código del programa para la manipulación de esos datos. Si una instancia no se está ejecutando, no se puede acceder a las bases de datos dentro de ella.

Hay dos tipos de bases de datos de SQL Server: bases de datos de sistema y de bases de datos de usuario. Cuando una instancia de SQL Server se instala por primera vez, cinco bases de datos del sistema se crean: **model, tempdb, master, msdb y resource**. Una instancia no se puede iniciar si cualquiera de sus bases de datos del sistema, excepto msdb esta inaccesible. Las bases de datos del usuario son creadas por desarrolladores y administradores de bases después que la instancia se ha instalado y el sistema de bases de datos ha iniciado. Estas son las bases de datos que almacenan la información de las organizaciones.

En el nivel físico, una base de datos de SQL Server está representada por un conjunto de archivos del sistema operativo que residen en el sistema de disco del servidor. Hay dos tipos de archivos de base de datos: el archivo de datos (**data file**) y el archivo de registro de transacciones (**transaction log file**).

Un archivo de datos es el repositorio central de información en una base de datos SQL, mientras que un archivo de registro de transacciones registra los cambios que se han aplicado a los datos. Este archivo es requerido por SQL Server para la recuperación del sistema.

Con Oracle, las cosas funcionan en la dirección inversa. Cuando Oracle se inicia, funciona igual que SQL en que una porción de la memoria del servidor se asigna para su funcionamiento. Esta área de memoria, conocido como el **Área Global de Sistema (SGA)**, se divide en una serie de estructuras diferentes. Junto con el espacio de memoria, una serie de procesos de fondo que también se inician para interactuar con el SGA. En

conjunto, el espacio de memoria y los procesos constituyen una instancia de Oracle. Sin embargo la base de datos Oracle todavía no está presente. De hecho, una instancia de Oracle podría estar funcionando perfectamente bien sin su base de datos en línea o incluso ser accesible. Una base de datos en Oracle es una colección de archivos de sistema operativo. A diferencia de SQL Server, una base de datos Oracle no representan a la agrupación lógica de los objetos, sino que es un único término genérico para una serie de archivos en el disco que principalmente tienen datos.

Los archivos que componen una base de datos de Oracle se pueden clasificar en tres tipos:

Archivo de datos (data file), archivo de rehacer (redo log file) y el **archivo de control (control file)**.

Los archivos de datos es donde residen todos los datos. Puede haber cualquier número de archivos de datos en una base de datos de Oracle. Archivos Rehacer son como los archivos de registro de transacciones de SQL Server que registra que cada cambio realizado a los datos y se utiliza para la recuperación del sistema. Los archivos de control son un tipo especial de archivo que contiene pequeñas piezas de información vital acerca de la base de datos. Sin este archivo, la instancia no será capaz de abrir la base de datos.

Aparte de los archivos de datos, archivos rehacer y los archivos de control, la base de datos contendrá también un **archivo de parámetros**, y un **archivo de contraseñas**, opcionalmente, **archivos de registro de archivado** (archive log file). Vamos a discutir acerca de cada tipo de archivos de base de datos Oracle en breve.

Cuando se inicia un sistema de Oracle, primero la instancia se crea en la memoria. La instancia a continuación, se conecta a la base de datos que residen en el disco y, finalmente, se abre la base de datos para la interacción del usuario. Cuando el sistema se apaga, la instancia se borrará de la memoria: todas las estructuras de memoria y los procesos se terminan, pero la base de datos todavía existe en el disco, aunque en un estado cerrado. Como se dijo anteriormente, es posible tener la instancia de Oracle que se ejecuta sin necesidad de abrir la base de datos – es una gran diferencia de SQL Server donde una instancia no puede comenzar sin primero tener sus bases de datos de sistemas en línea. Sin embargo,

como SQL Server, es imposible conectarse a una base de datos de Oracle, si la instancia no ha comenzado.

En general, la relación entre una instancia de Oracle y su base de datos es uno a uno. Las instalaciones de Oracle configuradas como **RAC (Real Application Cluster)** tendrán varias instancias que se ejecutan en diferentes máquinas que acceden a la misma base de datos en un disco compartido.

Entonces, ¿dónde está la agrupación lógica de los objetos de base de datos Oracle? En SQL Server, esta agrupación lógica es realizada por la propia base de datos. Para Oracle, se realiza a través de algo llamado **espacios de tablas (tablespaces)**. Un espacio de tablas de Oracle es una estructura lógica que agrupa a las tablas, vistas, indices y otros objetos de la base de datos. Por ejemplo, la base de datos Oracle de producción puede tener uno de tablas dedicado a la aplicación de recursos humanos y otro de tablas para la nómina. Cada espacio de tablas está físicamente representado por uno o más archivos de datos en el disco y forma parte de la base de datos.

4.2.3.2 BASES DE DATOS Y ESPACIO DE TABLAS

El proceso de creación de una base de datos en SQL Server es muy similar a la creación de un espacio de tablas en Oracle, ya que son muy similares en sus funciones. Cuando se crea una base de datos o de un espacio de tablas, el DBA (Administrador de Base de Datos) debe especificar un nombre. El DBA asigna uno o más archivos de datos a la base de datos o tablas de espacios y especifica el tamaño inicial y los incrementos de crecimiento de cada archivo.

Al igual que una base de datos de usuario de SQL Server se puede poner fuera de línea o de sólo lectura, también se puede en un espacio de tablas de usuario de Oracle. Y al igual que en SQL Server los archivos pueden ser de sólo lectura.

Sin embargo, las bases de datos y de tablas difieren entre sí en los siguientes puntos:

- En SQL Server, los archivos de datos pueden ser, lógicamente, agrupados en grupos de archivos. Los espacios de tablas de Oracle no tienen este concepto.

• En las bases de datos SQL Server, cada base de datos tendrá su propio registro de transacciones y las propiedades del archivo de registro deberá ser especificado durante la creación de bases de datos. Para Oracle, las transacciones de la base de datos es completa. Por consiguiente, no existe ninguna disposición para crear archivos de registro individuales para espacios de tablas.

4.2.3.3 LOS NOMBRES DE INSTANCIA

SQL Server y Oracle permiten ejecutar simultáneamente varias instancias del software servidor en el mismo equipo. Estos múltiples contextos de ejecución son totalmente independientes unos de otros.

En SQL Server, este mecanismo se activa a través del concepto de instancias. SQL Server puede funcionar tanto como una instancia nombrada o como una instancia predeterminada. La instancia predeterminada tiene el mismo nombre que el servidor de Windows que lo hospeda. Es posible ejecutar varias instancias nombradas en esa misma máquina. Cada instancia tendrá su propio conjunto de archivos binarios con algunos componentes comunes y compartidos entre todos.

Para Oracle, funciona de la misma manera. Cuando se instala Oracle, el DBA debe especificar un **nombre global de base de datos (Global Database Name)** y un **identificador del sistema (SID)**. La instancia y bases de datos son entidades completamente separadas en Oracle. Un SID por otra parte identifica la instancia asociada con la base de datos. En la mayoría de los casos una sola instancia se asocia a una única base de datos, el SID y el nombre de base de datos será el mismo. Los entornos de **Oracle Real Application Cluster** (RAC) son una excepción: RAC permite que múltiples instancias accedan a la misma base de datos alojada en un almacenamiento compartido; los nombres de instancia son diferentes del nombre de base de datos en estos casos. Sin embargo, al igual que un equipo de SQL Server, un servidor de base de datos Oracle no puede tener dos instancias en ejecución con el mismo SID.

Un DBA de SQL Server puede ejecutar la siguiente consulta para saber el nombre de la instancia a la que está actualmente conectado a:

```
1    SELECT @@SERVERNAME
```

Un DBA Oracle ejecutar consultas como la siguiente para obtener la instancia y el nombre de base de datos:

```
1  SELECT INSTANCE_NAME, HOST_NAME, VERSION,
   DATABASE_STATUS FROM V$INSTANCE;

2  SELECT NAME, DATABASE_ROLE, CREATED FROM V$DATABASE;
```

Una instancia de SQL Server dispondrá de cinco bases de datos de sistema (cuatro para las versiones anteriores de 2005) presente: master, model, msdb, tempdb y resource. Una base de datos Oracle necesita un mínimo de tres espacios de tablas de sistema para su funcionamiento: **SYSTEM**, **SYSAUX** y **TEMP**.

Las base de datos master y resource son los repositorios centrales de toda la información de SQL Server para gestionar las necesidades de sí mismo. Para Oracle, el espacio de tablas del sistema es el equivalente de la base de datos master. El espacio de tablas SYSTEM contiene el **diccionario de datos**, que son los metadatos de Oracle sobre sí misma. El diccionario de datos se puede comparar con la base de datos de recursos de SQL.

Para una instancia de SQL Server, la base de datos model es la "plantilla" que se utiliza para crear cada nueva base de datos en esa instancia. Para Oracle, no existe tal modelo, pero cuando se crea un espacio de tablas, puede especificar si será un espacio de tablas permanente o de cualquier otro tipo como un espacio de tablas TEMP o UNDO.

V.- MATERIALES Y METODOS

Universo

El Universo de esta investigación son las bases de datos en general, y usaremos una muestra de 2 bases de datos creadas para realizar el estudio comparativo, una la Base de Datos "Compañía" (Proyectos realizados en una Compañía) para probar las sentencias de SQL 2005 y una Base de Datos "HR" (Historia laboral de Empleados) para comprobar las sentencias en ORACLE. El tipo de muestra es no probabilística.

Las versiones de software que se utilizarán serán en el Caso de SQL, el SQL 2012, por lo que me referiré a este como SQL server y en el caso de ORACLE he utilizado la versión ORACLE 11g. Ya que son las últimas versiones de ambos en la fecha de ejecución del proyecto, que es el 2013.

Técnicas de Recopilación de datos

Las técnicas de recopilación de datos utilizada es la Observación del comportamiento en la práctica de cada una de las Bases de Datos creadas en SQL 2005 y ORACLE respectivamente.

En cada una de las Bases de Datos creadas, se probará de manera práctica, las características del DDL, DML y DCL para observar el comportamiento de cada una de las sentencias.

Adicionalmente, también se realizarán cuadros de contrastación de resultados.

En cuanto al método utilizado es el Método inductivo, ya que hemos hecho una transición de lo particular a lo general, se ha usado la generalización inductiva, porque la investigación se basa en hechos reales para lograr la evidencia.

Técnicas Estadísticas

En cuanto a las técnicas estadísticas usadas tenemos que esta investigación es de **tipo cualitativo**, es decir que tiene las características de una investigación de esta naturaleza, como son: Estudio de casos, Generalizar conclusiones de casos, empleo del método inductivo a diferencia de las investigaciones cuantitativas que utilizan el método deductivo, estudian poblaciones o muestras y que emplean métodos estadísticos para analizar los datos y generar datos numéricos.

Por lo tanto esta investigación no utiliza técnicas estadísticas por ser una investigación cualitativa.

VI.- RESULTADOS

6.1 BASES DE DATOS COMPARADAS

6.1.1 SQL SERVER

En SQL se utilizó para la comparación, motivo de la presente investigación, la Base de Datos Compañía, conformada por las siguientes tablas y datos.

EMPLEADO

Nombrep	inic	Apellido	NSS	Fechan	Direccion	sexo	Salario	NSS SUPER	ND
José	B	Silva	123456789	09-01-55	Fresnos ...	M	30000	333445555	5
Federico	T	Vizcarra	333445555	08-12-45	Valle 638,...	M	40000	888665555	5
Alicia	J	Zapata	999887777	19-07-58	Catillo 3321,	F	25000	987654321	4
Jazmin	S	Valdés	987654321	20-06-31	Bravo 291,..	F	43000	888665555	4
Ramón	K	Nieto	666884444	15-09-52	Espiga 875,	M	38000	333445555	5
Josefa	A	Esparza	453453453	31-07-62	Rosas 5631,	F	25000	333445555	5
Ahmed	V	Jabbar	987987987	29-03-59	Dalias 980,	M	25000	987654321	4
Jaime	E	Botello	888665555	10-11-27	Sorgo 450,	M	55000	Nulo	1

DEPARTAMENTO

Nombred	Númerod	NSSGTE	Fechalnicgte
Investigación	5	333445555	22-05-78
Administración	4	987654321	01-01-85
Dirección	1	888665555	19-06-71

LUGARES_DEPTOS

Númerod	Lugard
1	Higueras
4	Santiago
5	Belén
5	Sacramento
5	Higueras

PROYECTO

Nombrep	Númerop	Lugarp	Numd
ProductoX	1	Belén	5
ProductoY	2	Sacramento	5
ProductoZ	3	Higueras	5
Automatización	10	Santiago	4
Reorganización	20	Higueras	1
Nuevasprestaciones	30	Santiago	4

TRABAJA_EN

NSSE	Nump	Horas
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	nulo

DEPENDIENTE

NSSE	Nombre_dependiente	sexo	fechan	Parentesco
333445555	Alicia	F	05-04-76	Hija
333445555	Teodoro	M	25-10-73	Hijo
333445555	Jobita	F	03-05-48	Cónyuge
987654321	Abdiel	M	29-02-32	Cónyuge
123456789	Miguel	M	01-01-78	Hijo
123456789	Alicia	F	31-12-78	Hija
123456789	Elizabeth	F	05-05-57	Cónyuge

Figura 6.1 Tablas y datos definidos en la Base de Datos Compañía

Fuente: Autor

6.1.2 ORACLE

En ORACLE se utilizó para la comparación estudiada, la Base de Datos HR, cuyo esquema físico es el siguiente.

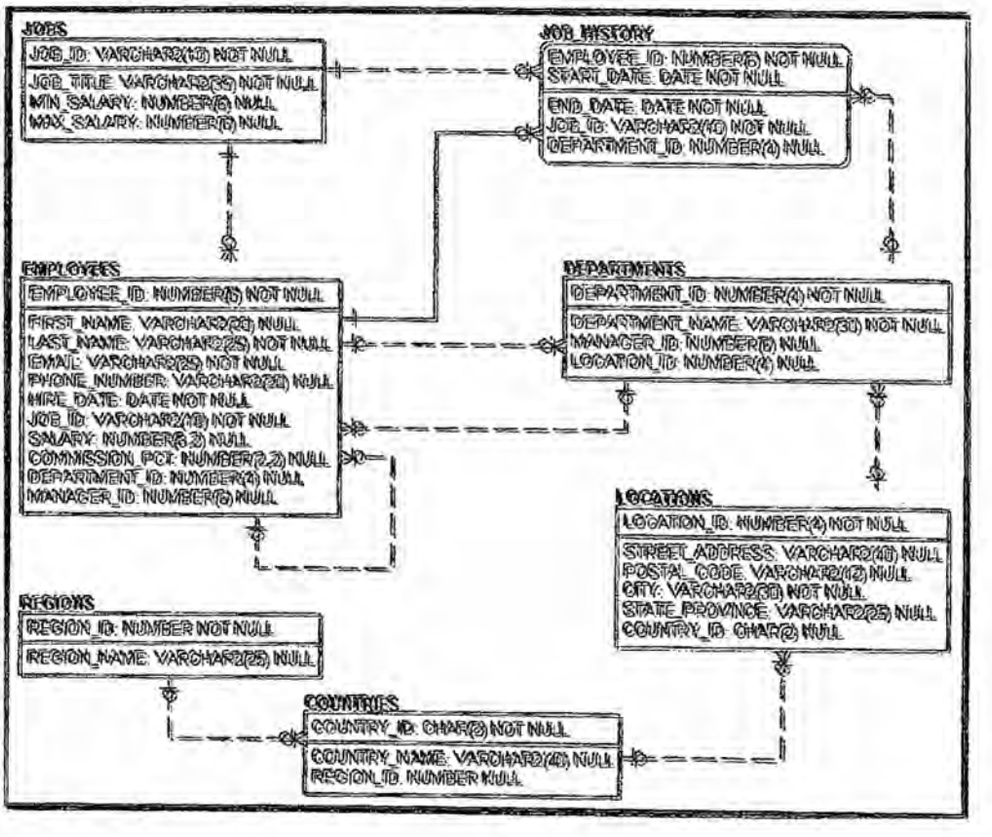

Figura 6.2 Esquema de la Base de Datos HR. Fuente: ORACLE

Resultado:

Se mostró los esquemas de la Base de Datos "Compañía" para probar las sentencias de SQL 2005 y la Base de Datos "HR" para comprobar las sentencias en ORACLE, "HR" es una Base de Datos de prueba que proporciona el ORACLE.

6.2 LENGUAJE DE DEFINICIÓN DE DATOS (DDL)

6.2.1 CREACION DE ESQUEMAS Y TABLAS

SQL Server

A continuación mostraremos el código para crear la Base de Datos y las tablas de la Base de Datos Compañía en SQL 2005.

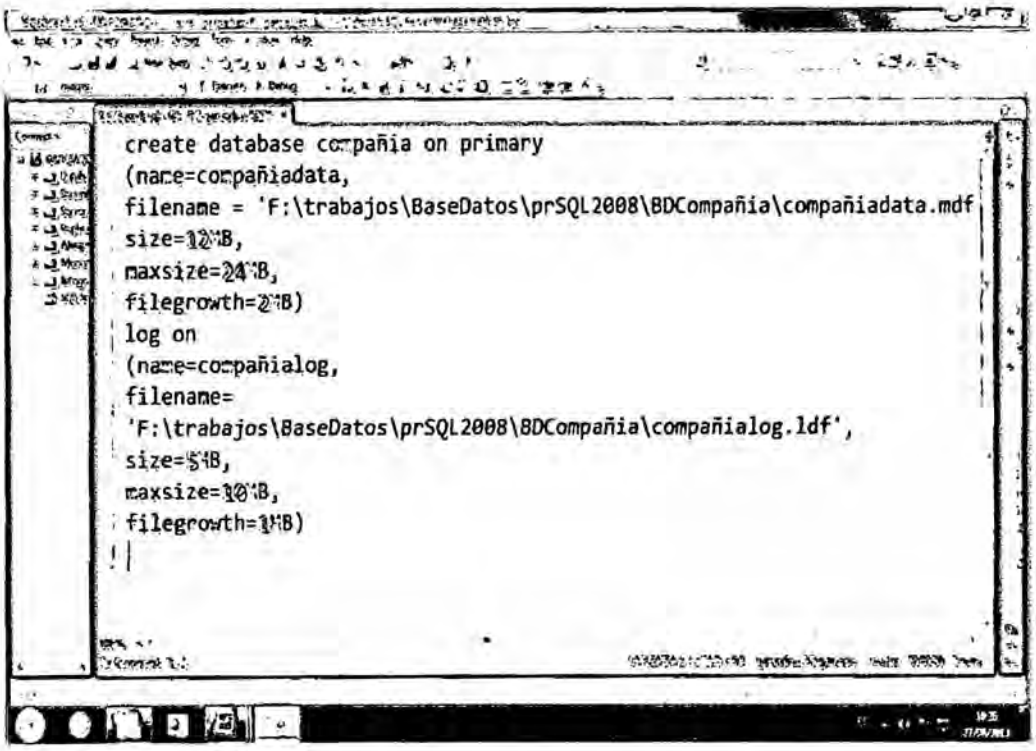

Figura 6.3 Sentencias para la creación de la Base de Datos Compañía

Fuente: Autor

```
CREATE TABLE EMPLEADO (
       nombrep          varchar(15) NOT NULL,
       inic             char,
       apelli           varchar(15) NOT NULL,
       nss              char (9)    NOT NULL PRIMARY KEY,
       fechan           datetime,
       direccion        varchar(30),
       sexo             char,
       salario          decimal(10,2),
       nsssuper         char(9),
       nd               int         NOT NULL
 )
CREATE TABLE DEPARTAMENTO (
       nombred          varchar(15)NOT NULL,
       numerod          int         NOT NULL PRIMARY KEY,
       nssgte           char(9)     NOT NULL,
       fechainicgte     datetime
)

CREATE TABLE LUGARES_DEPTOS (
         numerod        int         NOT NULL,
        lugard          varchar(15) NOT NULL
         PRIMARY KEY (numerod, lugard))

CREATE TABLE PROYECTO (
       nombrepr         varchar(20) NOT NULL,
       numerop          int          NOT NULL PRIMARY KEY,
       lugarp           varchar(15),
       numd             int          NOT NULL,
  )

CREATE TABLE EMPLEADO_PROYECTO (
       nsse             char(9)     NOT NULL,
       nump             int         NOT NULL,
       horas            decimal(3,1) NOT NULL
)
ALTER TABLE EMPLEADO_PROYECTO
       ADD PRIMARY KEY (nsse, nump)

CREATE TABLE DEPENDIENTE (
      nsse               char(9)   NOT NULL,
      nombre_dependiente varchar(15) NOT NULL,
      sexo               char,
      fechan             datetime,
      parentesco         varchar(8)
      PRIMARY KEY (nsse,nombre_dependiente)
)
```

Figura 6.4 Creación de la estructura de la Base de Datos Compañía

Fuente: Autor

```
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('José','B','Silva','123456789','09/01/1955','Fresnos
731,Higueras,MX','M',30000,'333445555',5)
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('Federico','T','Vizcarra','333445555','08/12/1945','Valle 638,
Higueras,MX','M',40000,'888665555',5)
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('Alicia','J','Zapata','999887777','19/07/1958','Catillo,3321,
Sucre, MX','F',25000,'987654321',4)
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('Jazmin','S','Valdes','987654321','20/06/1931','Bravo 291,
Belén,MX','F',43000,'888665555',4)
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('Ramón','K','Nieto','666884444','15/09/1952','Espiga 875,
Heras,MX','M',38000,'333445555',5)
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('Josefa','A','Esparza','453453453','31/07/1962','Rosas 5631,
Higueras,MX','F',25000,'333445555',5)
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('Ahmed','V','Jabbar','987987987','29/03/1959','Dalias 980,
Higueras,MX','M',25000,'987654321',4)
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,salario,nss
super,nd) values
('Jaime','E','Botello','888665555','10/11/1927','Sorgo 450,
Higueras,MX','M',55000,'123456789',1)
insert into DEPARTAMENTO(nombred,numerod,nssgte,fechainicgte) values
('Investigación',5,'333445555','22/05/1978')
insert into DEPARTAMENTO(nombred,numerod,nssgte,fechainicgte) values
('Administración',4,'987654321','01/01/1985')
insert into DEPARTAMENTO(nombred,numerod,nssgte,fechainicgte) values
('Dirección',1,'888665555','19/06/1971')
insert into LUGARES_DEPTOS(numerod,lugard) values (1,'Higueras')
insert into LUGARES_DEPTOS(numerod,lugard) values (4,'Santiago')
        ….
```

Figura 6.5 Inserción de datos a la Base de Datos Compañía

Fuente: Autor

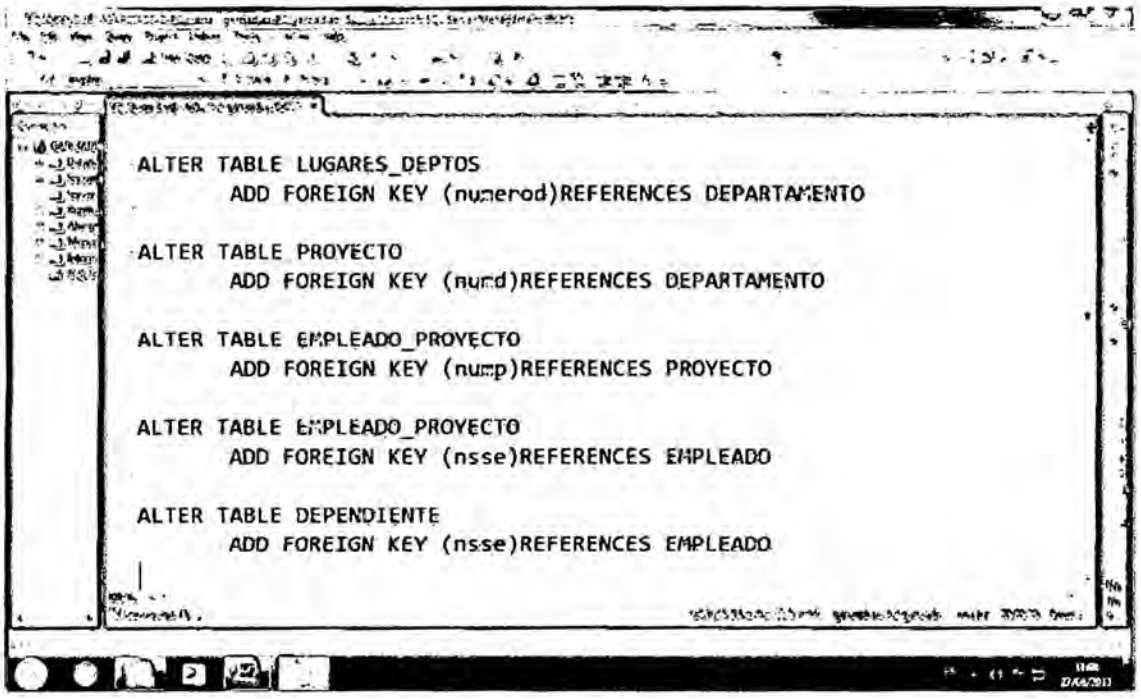

Figura 6.6 Creación de claves foráneas a la Base de Datos Compañía

Fuente: Autor

ORACLE

Esquemas de Base de Datos

En Oracle, un esquema es una colección de objetos de la base de datos pertenecientes a un usuario específico.

Por ejemplo, si creamos el usuario ventas, y queremos que exista el esquema ventas debemos hacer lo siguiente:

1. Debemos asignarle una cuota al usuario ventas en algún tablespace.
2. Debemos asignarle privilegios al usuario ventas para que pueda crear sus objetos (tablas).
3. Finalmente, el usuario ventas debe crear sus objetos.

Después de realizar los tres pasos anteriores podemos decir que tenemos el esquema ventas.

Crear el esquema ventas con tres tablas:

1. Tabla de artículos: ARTICULO
2. Tabla de ventas: VENTA
3. Tabla de usuarios: USUARIO

```
connect / as sysdba

create user ventas
identified by ventas;

grant connect, resource to ventas;
```

Figura 6.7 Creación del usuario y asignación de cuota y privilegios

Fuente: Autor

```
connect ventas/ventas
create table articulo(
art_id number(5) primary key,
art_nombre varchar2(50) not null,
art_precio number(8,2) not null
);

create table usuario(
usu_id number(5) primary key,
usu_nombre varchar2(50) not null,
usu_usuario varchar2(15) not null,
usu_clave varchar2(15) not null
);

create table venta(
ven_id number(5) primary key,
ven_fecha date not null,
usu_id number(5) not null,
art_id number(5) not null,
ven_cant number(5) not null,
ven_precio number(8,2) not null
);

alter table venta
add constraint fk_venta_articulo
foreign key( art_id )
references articulo;

alter table venta
add constraint fk_venta_usuario
foreign key( usu_id )
references usuario;
```

Figura 6.8 Creación de los objetos. Fuente: Autor

Resultado:

Existen diferencias en cuanto a la creación de Base de Datos, en SQL 2005 se asigna la ubicación física de la Base de Datos y crea un archivo con extensión mdf y log. Y en Oracle se crea un Usuario y se le asigna en algún tablespace y luego se asignan privilegios para crear tablas.

En cuanto a la creación de tablas, claves primarias y foráneas no existen diferencias entre ambos.

6.2.2 IMPLEMENTAR CONSTRAINTS

6.2.2.1 CHECK CONSTRAINTS

SQL Server

Ejemplo: Implementar un check constraint que verifique que los salarios de los empleados sean mayores que 24000 pesetas.

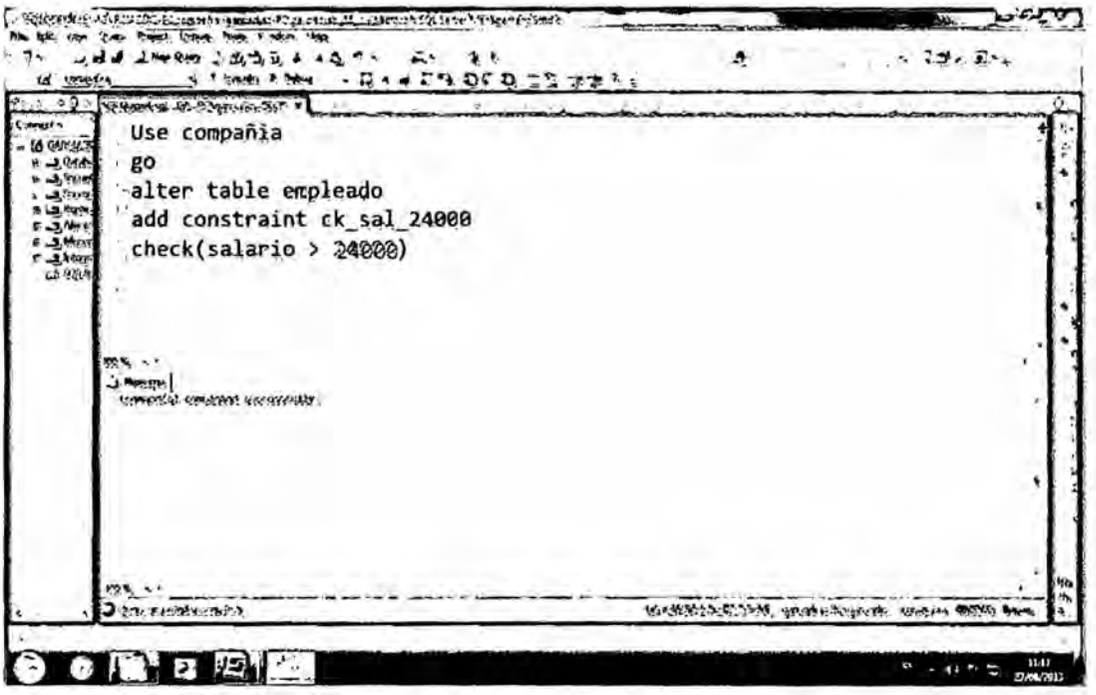

Figura 6.9 Creación de un check constraints. Fuente: Autor

ORACLE

Ejemplo: Tenemos un constraint que chequea el campo sexo.

```
alter table clientes
add constraint ck_sexo
check (sex in ('M','F', 'm','f')),
```

Resultado:

No existen diferencias en el comando check constraint, en ambos SGBD.

6.2.2.2 DEFAULT (VALORES POR DEFECTO)

SQL Server

Ejemplo: Asignar un valor por defecto a la columna nsssuper de la tabla empleado haciendo que el código por defecto sea 333445555.

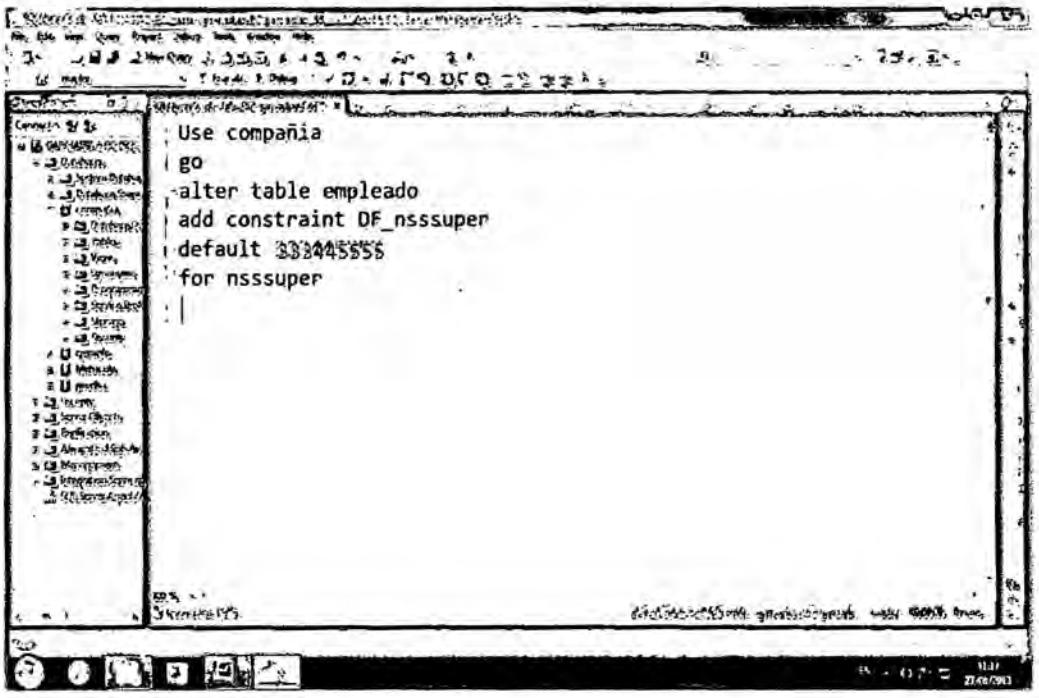

Figura 6.10 Creación de un default. Fuente: Autor

ORACLE

Ejemplo: El salario por defecto para cualquier Empleado debe ser 1000.

```
Alter table employees
Modify ( salary default 1000)
```

Resultado:

Hay diferencias en la sintáxis para otorgar un valor por defecto en ambos Gestores de Bases de Datos ya que en Oracle es necesario ingresar el comando Modify.

6.2.3 MODIFICACION DE TABLAS (ALTER TABLE)

6.2.3.1 MODIFICAR COLUMNAS

SQL Server

Ejemplo: Modifique el campo lugarp de la tabla Proyecto para asignarle como tipo de datos varchar con longitud 20 y que no permita valores NULL

```
alter table proyecto
alter column lugarp varchar(20) not null
```

Ejemplo: Agregue un campo presupuesto de tipo money a la tabla Proyecto.

```
alter table proyecto
add presupuesto money
```

Ejemplo: Elimine la columna presupuesto de la tabla Proyecto.

```
alter table proyecto
drop column presupuesto
```

ORACLE

Ejemplo: Modifique la tabla Countries para adicionar la columna zon de tipo char.

```
alter table COUNTRIES add zon char(1);
```

Ejemplo: Modifique la tabla Countries para modificar la columna zon de tipo char(1) a char(2)

```
alter table COUNTRIES modify zon char(2);
```

Ejemplo: Modifique la tabla Countries para eliminar la columna zon de tipo char(1) a char(2)

```
alter table COUNTRIES modify drop column zon;
```

Resultado:

Hay diferencias en los comandos para modificación y eliminación de columnas en ambos Gestores de Bases de Datos ya que en Oracle es necesario ingresar el comando Modify, pero no hay diferencias para adicionar columnas.

6.3 LENGUAJE DE MANIPULACION DE DATOS (DML)

6.3.1 INSERT

SQL Server

Ejemplo: Insertar un registro a la tabla Empleado.

```
insert into
EMPLEADO(nombrep,inic,apellido,nss,fechan,direccion,sexo,
salario,nsssuper,nd) values
('José','B','Silva','123456789','09/01/1955','Fresnos
731,Higueras,MX','M',30000,'333445555',5)
```

ORACLE

Ejemplo: Insertar un registro a la tabla Departments.

```
insert into hr.departments(department_id,
department_name, manager_id, location_id) values
 (70,'public relations',100, 1700);
```

Resultado:

No hay diferencias en el comando Insert en ambos lenguajes algebraicos.

6.3.2 UPDATE

SQL Server

Ejemplo: Actualizar la dirección del Empleado de nss 123456789

```
UPDATE EMPLEADO
SET direccion = 'Turin 654, Belén,MX'
WHERE nss = '123456789'
```

ORACLE

Ejemplo: Actualizar el department_name de la tabla Departments.

```
update hr.departments
set department_name='Sistemas'
where department_id=60;
```

Resultado:

No hay diferencias en el comando Update para actualizar filas en ambos lenguajes algebraicos.

6.3.3 DELETE

SQL Server

Ejemplo: Eliminar de la tabla Dependiente, a las mujeres

```
DELETE DEPENDIENTE
WHERE sexo = 'F'
```

ORACLE

```
delete from departments
where department_name ='Finance';
```

Resultado:

Hay una pequeña diferencia de sintaxis en el comando Delete, ya que en SQL no requiere del from.

6.3.4 SELECT

6.3.4.1 CONSULTAS BASICAS

SQL Server

Ejemplo: Obtener la fecha de nacimiento y la dirección del empleado cuyo nombre es José B. Silva.

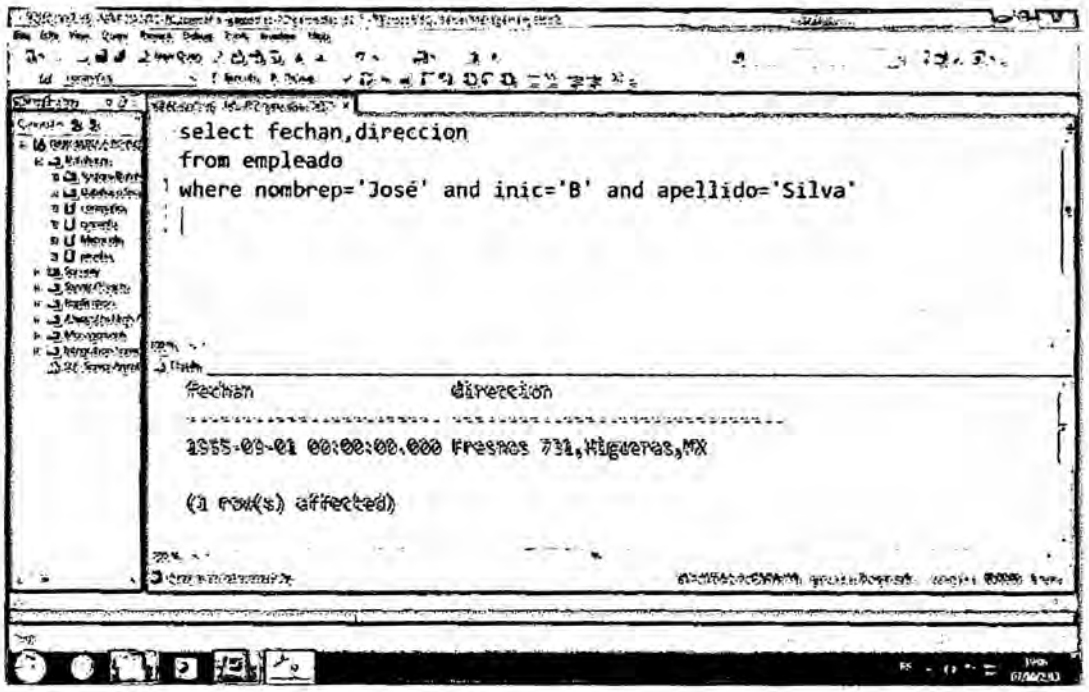

Figura 6.11 Consulta Básica. Fuente: Autor

Ejemplo: obtener el nombre y la dirección de todos los empleados que trabajan para el departamento de 'Investigación'.

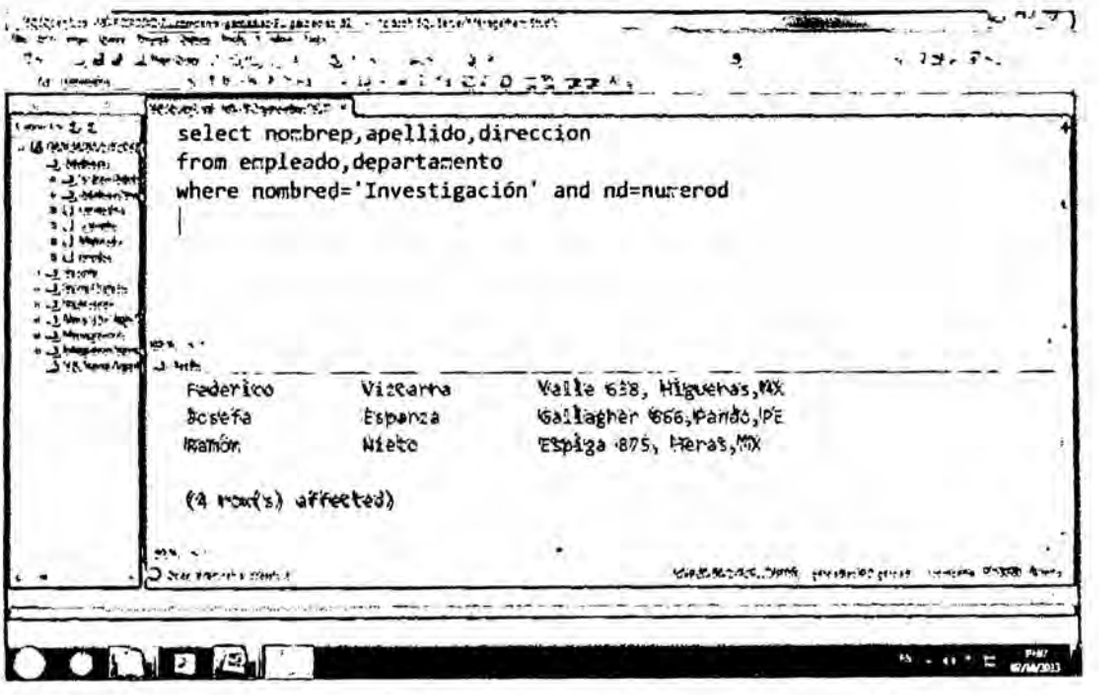

Figura 6.12 Consulta multitabla. Fuente: Autor

Ejemplo: Obtener el nombre y la dirección de todos los empleados que trabajan para el departamento de 'investigación'.

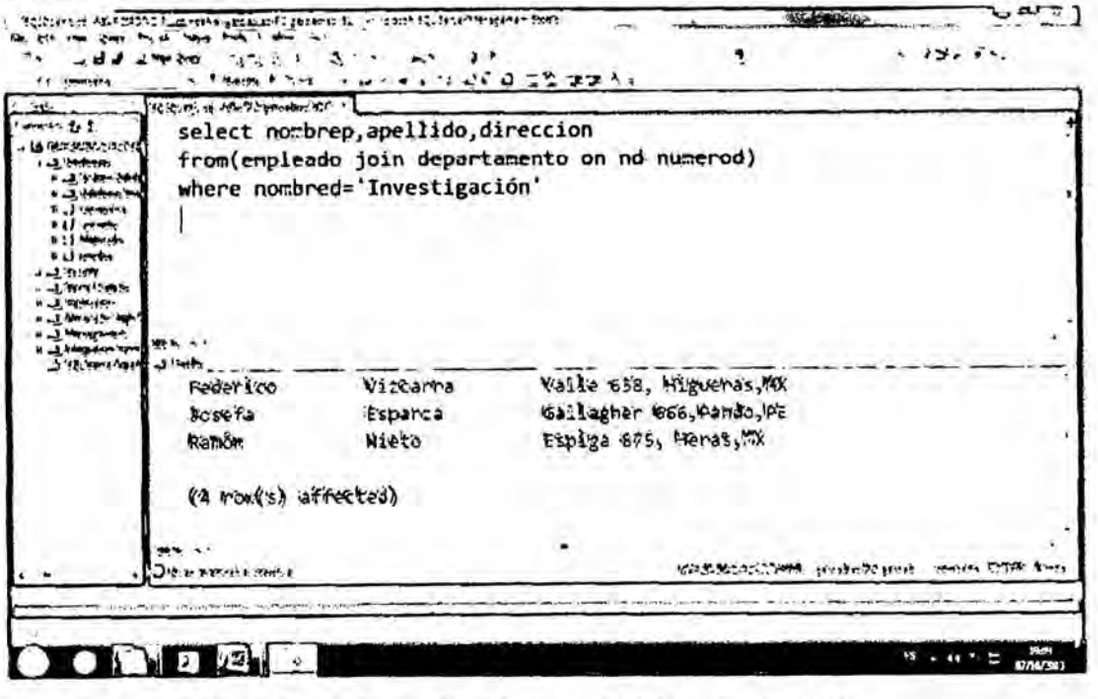

Figura 6.13 Consulta multitabla usando join. Fuente: Autor

Ejemplo: para cada proyecto ubicado en 'Santiago', listar el número del proyecto, el número del departamento controlador y el apellido, la dirección y la fecha de nacimiento del gerente de ese departamento.

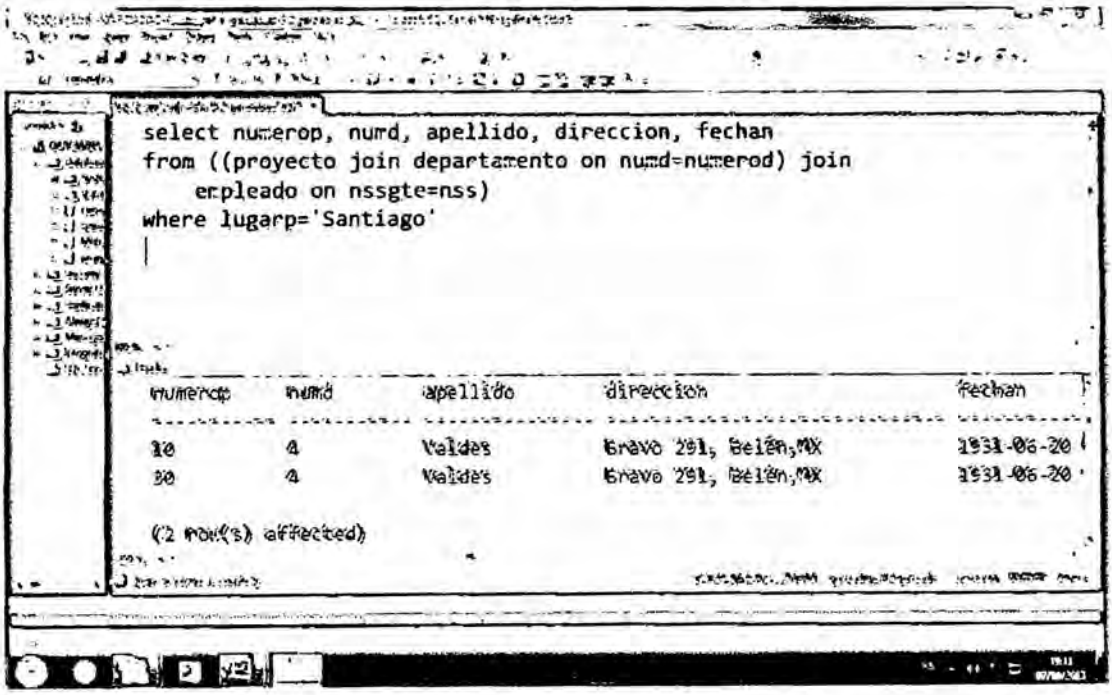

Figura 6.14 Consulta multitabla usando join. Fuente: Autor

ORACLE

Ejemplo: Mostrar los Empleados que ganen menos de 1000.

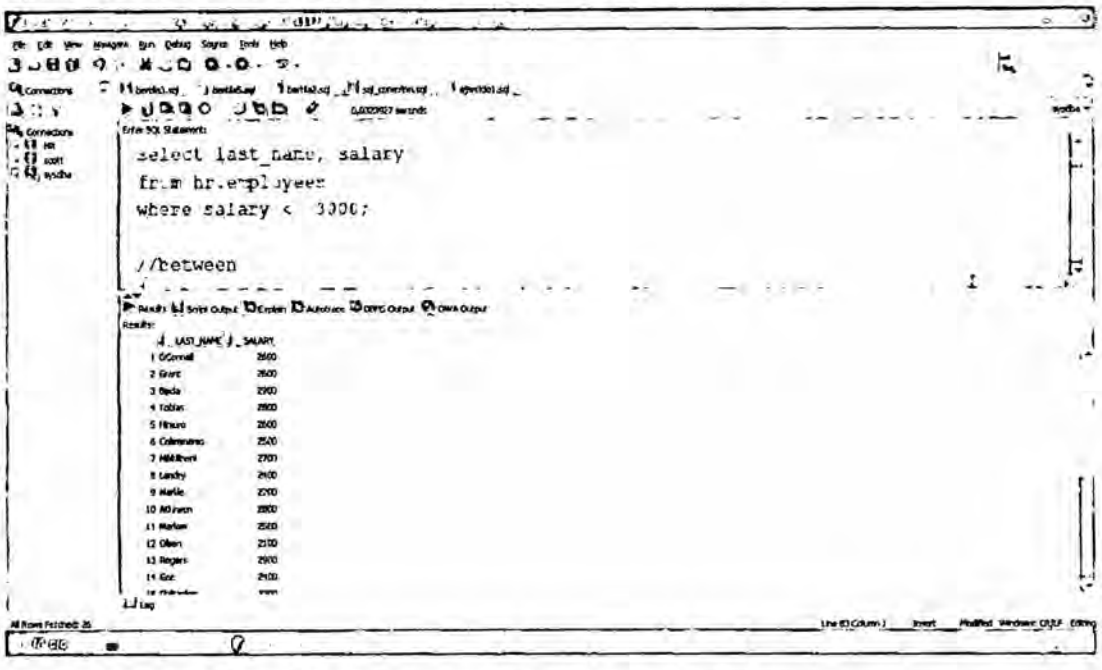

Figura 6.15 Consulta Básica. Fuente: Autor

Ejemplo: Mostrar nombre de Departamento y ciudad de los Departamentos. ,

Figura 6.16 Consulta multitabla. Fuente: Autor

Ejemplo: Mostrar nombre de Departamento y ciudad de los Departamentos.

Figura 6.17 Consulta multitabla usando natural join. Fuente: Autor

Ejemplo: Mostrar Apellidos y Departamento de origen de los Empleados.

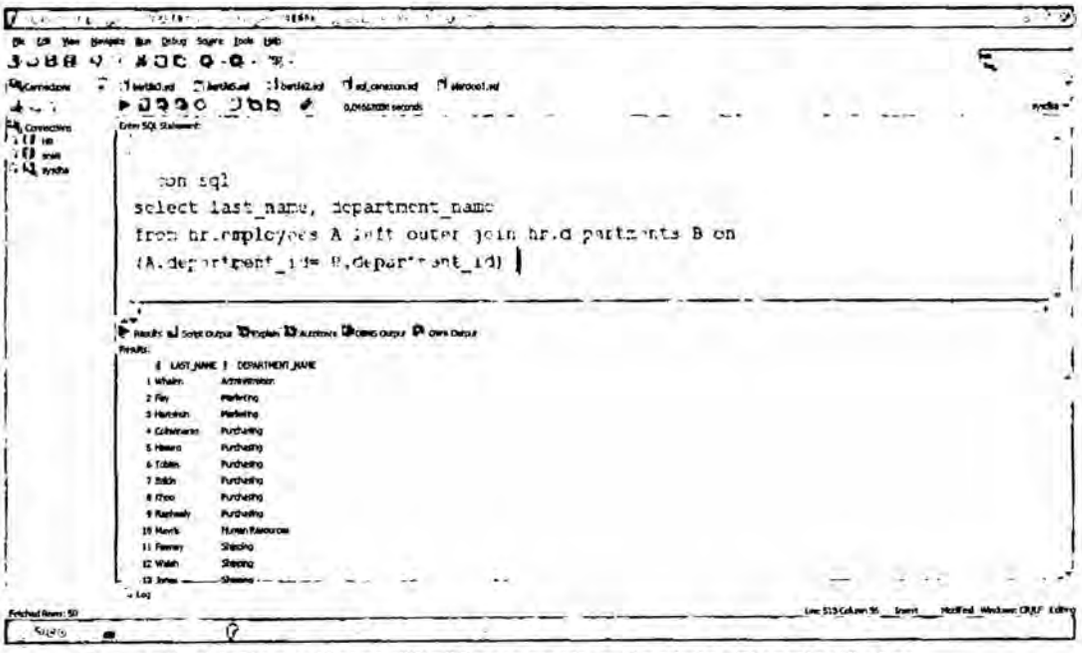

Figura 6.18 Consulta multitabla usando outer join. Fuente: Autor

Resultado

No hay diferencias en cuanto a Sentencias Básicas en ambos SGBD.

6.3.4.2 MANEJO DE NOMBRES DE ATRIBUTOS AMBIGUOS

SQL Server

Ejemplo: Para cada empleado, obtener su nombre de pila y apellido y el nombre de pila y apellido de su supervisor inmediato.

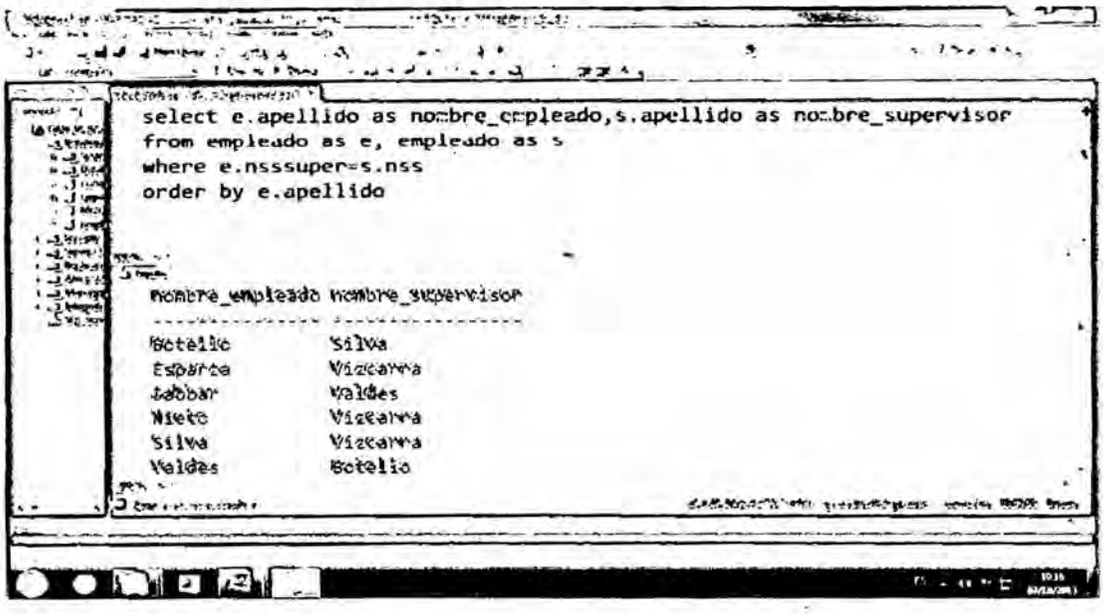

Figura 6.19 Consulta multitabla de una tabla recursiva. Fuente: Autor

ORACLE

Ejemplo: Mostrar datos de los Empleados y su Jefe.

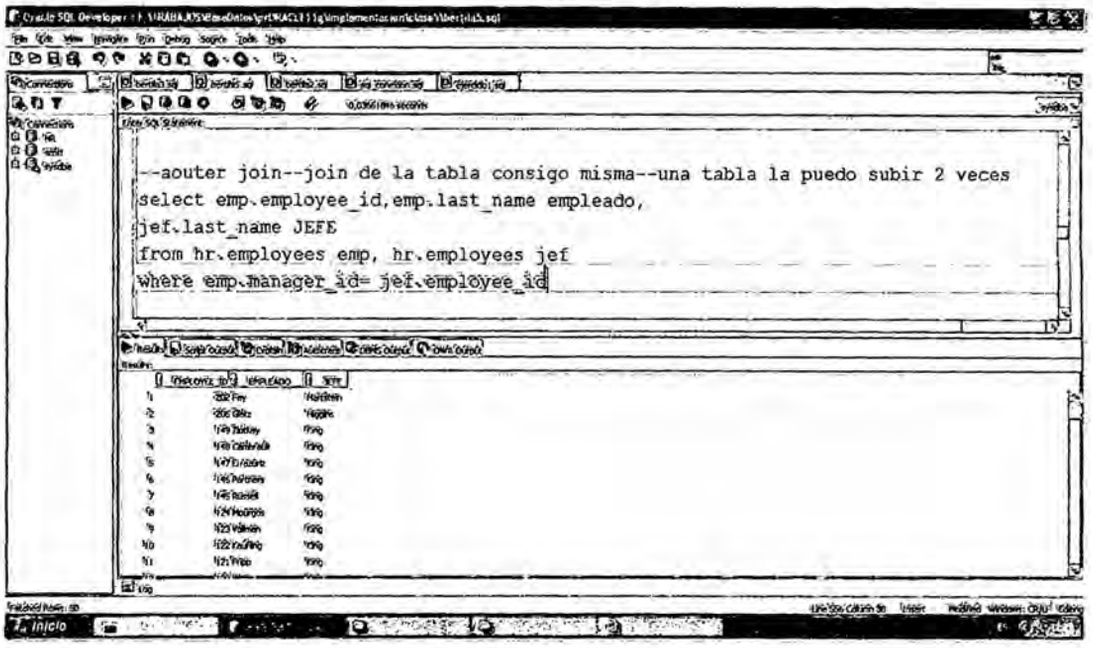

Figura 6.20 Consulta multitabla de una tabla recursiva. Fuente: Autor

Resultado:

No hay diferencias en cuanto al manejo de nombres de atributos ambiguos en ambos gestores de bases de datos.

6.3.4.3 CLAUSULAS WHERE NO ESPECIFICADAS Y EMPLEO DE *

SQL Server

Ejemplo: Obtener los valores de todos los atributos de las tuplas de empleados que pertenecen al departamento 5.

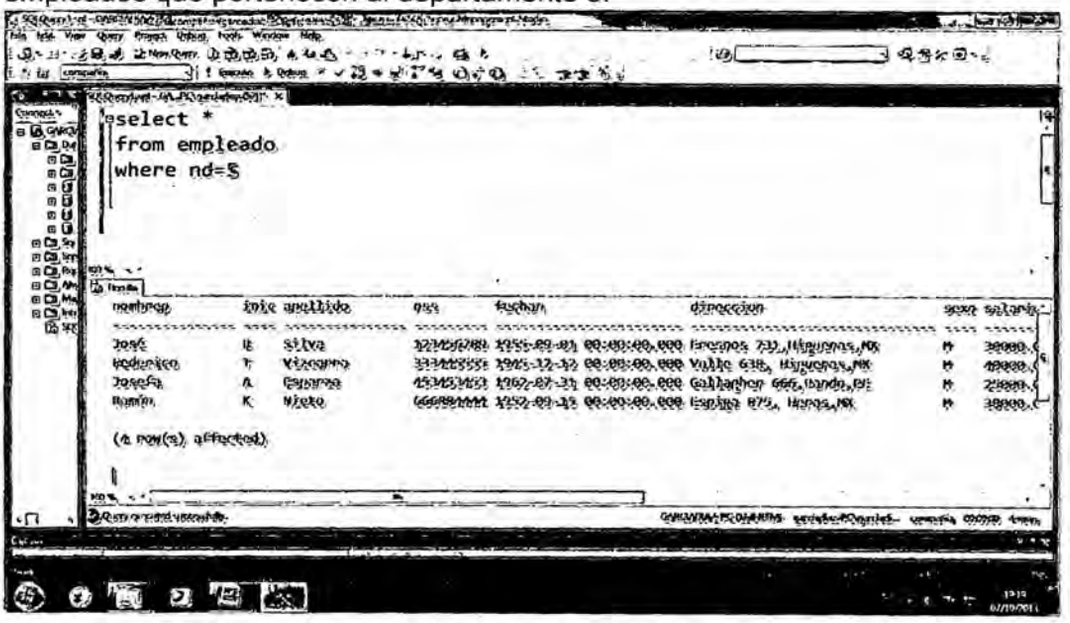

Figura 6.21 Consulta usando la clausula Where. Fuente: Autor

Ejemplo: Obtener todos los atributos de las tuplas de empleados que pertenecen al departamento 'Investigación'.

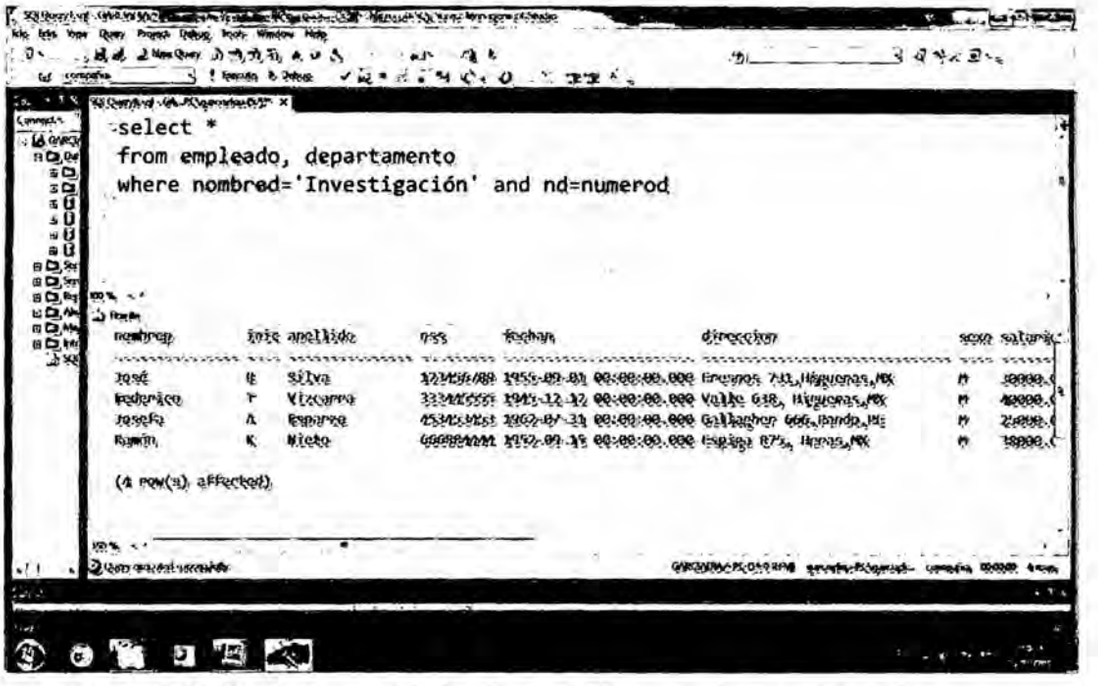

Figura 6.22 Consulta usando la clausula Where y el operador and.
Fuente: Autor

ORACLE

Ejemplo: Mostrar los Empleados con fecha de contrato 1996.

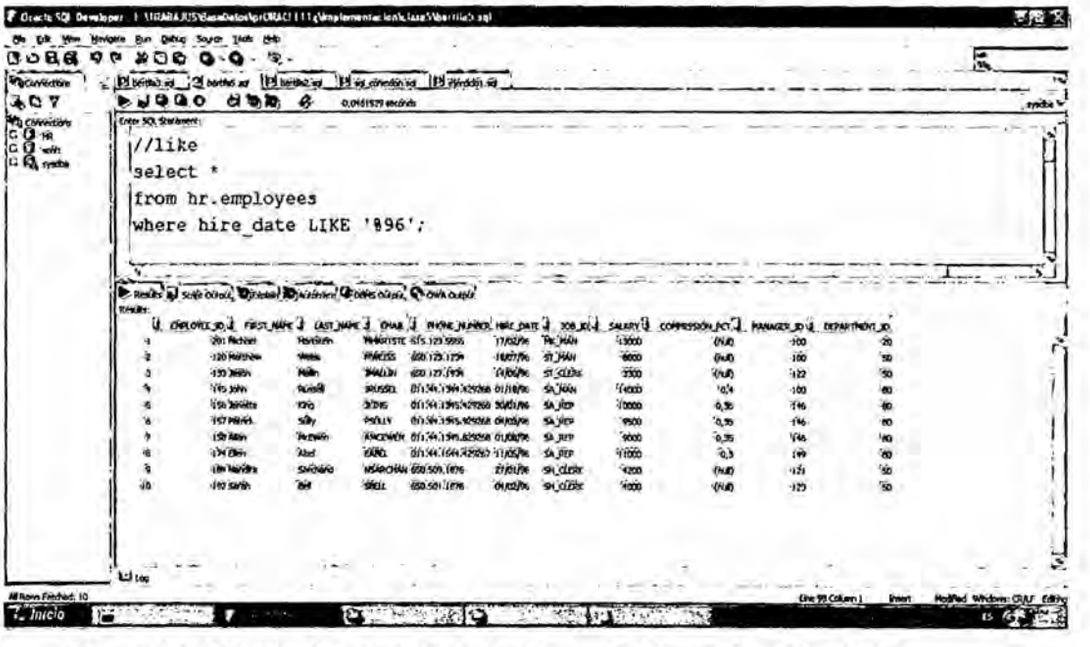

Figura 6.23 Consulta usando * y la clausula Where. Fuente: Autor

Resultado:

No hay diferencias en cuanto al manejo de clausulas where no especificadas y empleo de *, en ambos gestores de bases de datos.

6.3.4.4 TABLAS COMO CONJUNTOS EN SQL

SQL Server

Ejemplo: Obtener el salario de todos los empleados.

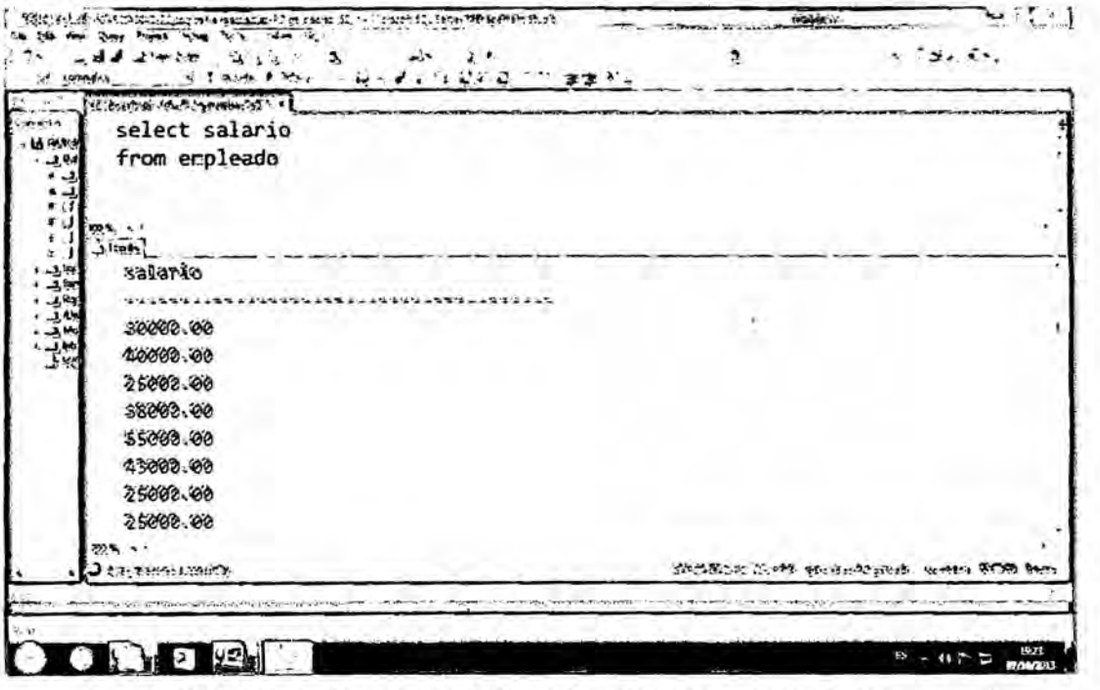

Figura 6.24 Consulta usando la tabla Empleado. Fuente: Autor

Ejemplo: Obtener salarios distintos de empleados.

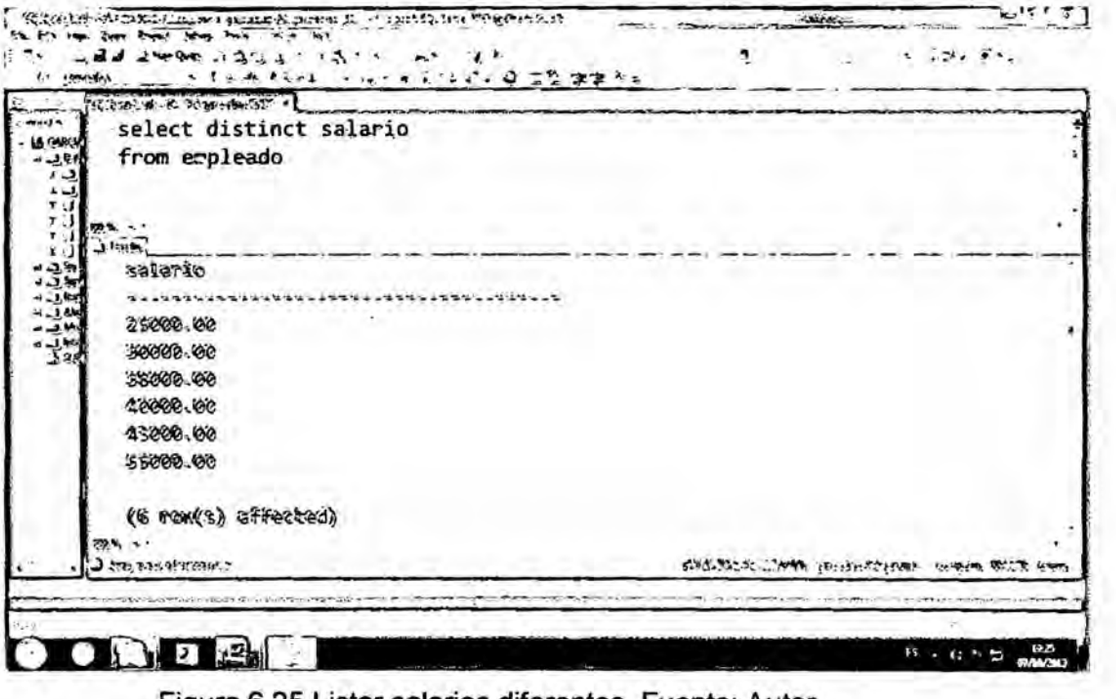

Figura 6.25 Listar salarios diferentes. Fuente: Autor

ORACLE

Ejemplo: Listar el código de Departamento de la tabla Empleado.

Figura 6.26 Mostrar con filas duplicadas. Fuente: Autor

Ejemplo: Listar códigos diferentes de Departamento de la tabla Empleado

Figura 6.27 Mostrar con filas diferentes. Fuente: Autor

Resultado:

No hay diferencias en cuanto a tablas como conjuntos en ambos lenguajes algebraicos.

6.3.4.5 CONSULTAS ANIDADAS Y COMPARACIONES DE CONJUNTOS

SQL Server

Ejemplo: Preparar una lista con todos los números de los proyectos en los que participa un empleado de apellido 'Vizcarra', sea como trabajador o como Gerente del departamento que controla el proyecto.

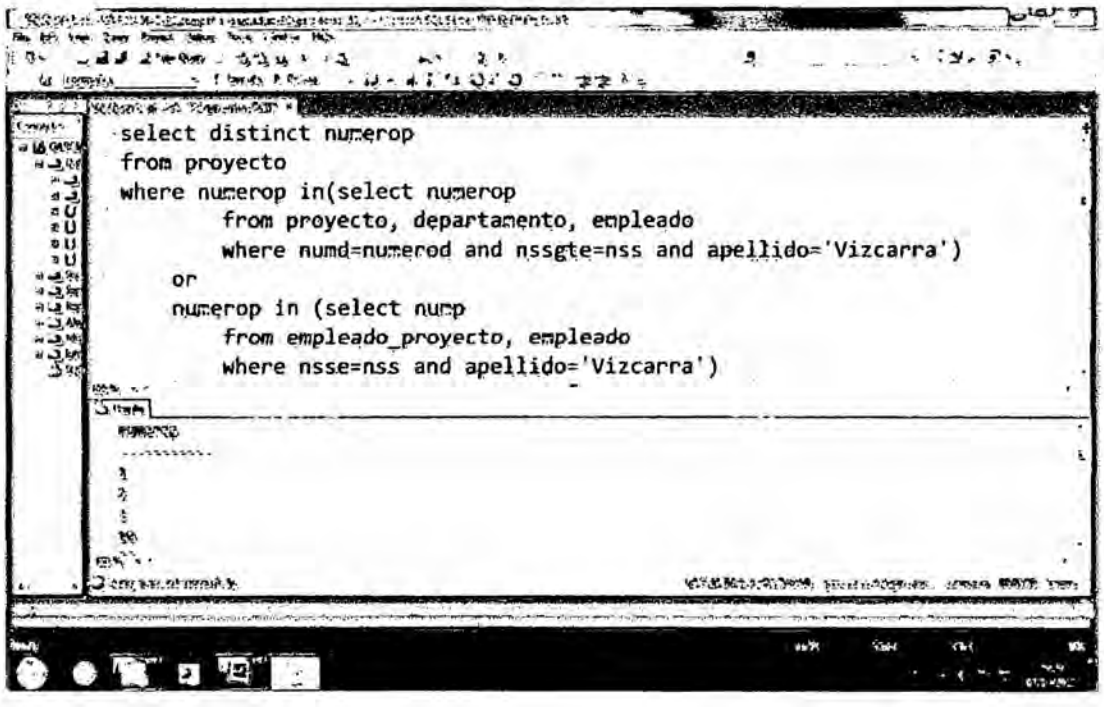

Figura 6.28 Muestra consultas anidadas. Fuente: Autor

Ejemplo: Obtener los nombres de los empleados cuyo salario es mayor que el de todos los empleados del departamento 5.

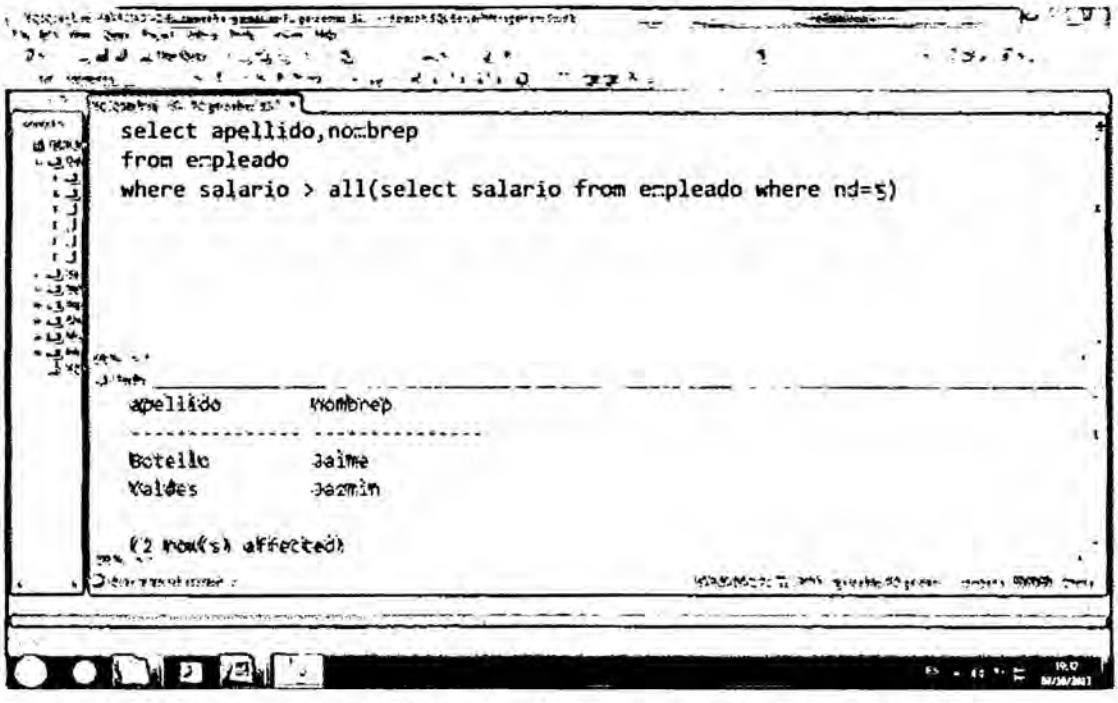

Figura 6.29 Consulta usando la palabra ALL con operador >. Fuente: Autor

Ejemplo: Obtener el nombre de todos los empleados que tienen un dependiente con el mismo sexo que el empleado.

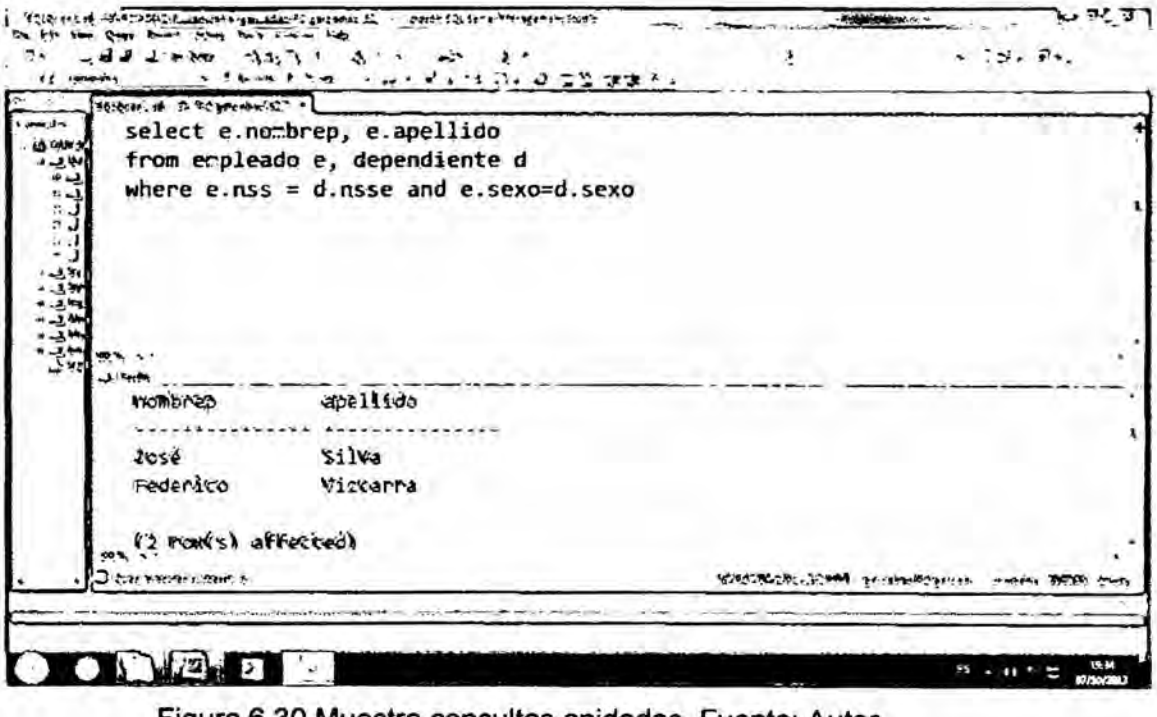

Figura 6.30 Muestra consultas anidadas. Fuente: Autor

ORACLE

Ejemplo: Realizar una consulta que muestre el empleado con el menor salario.

Figura 6.31 Muestra un subquery. Fuente: Autor

Ejemplo: Mostrar los nombres de los empleados, cuyo salario sea menor a aquellos cuyo job_id = 'IT_PROG' , pero cuyo job_id <> 'IT_PROG'.

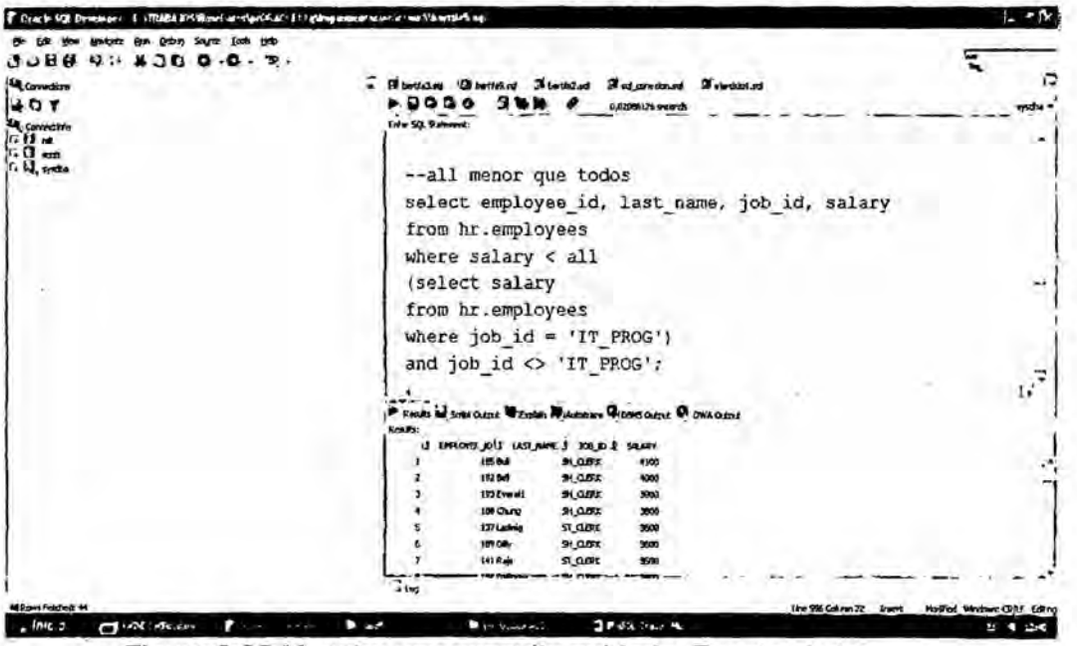

Figura 6.32 Muestra una consulta anidada. Fuente: Autor

Resultado:

No hay diferencias en cuanto a consultas anidadas y comparaciones de conjuntos en ambos gestores de bases de datos.

6.3.4.6 LA FUNCION EXISTS

SQL Server

Ejemplo: Obtener el nombre de todos los empleados que tienen un dependiente con el mismo sexo que el empleado.

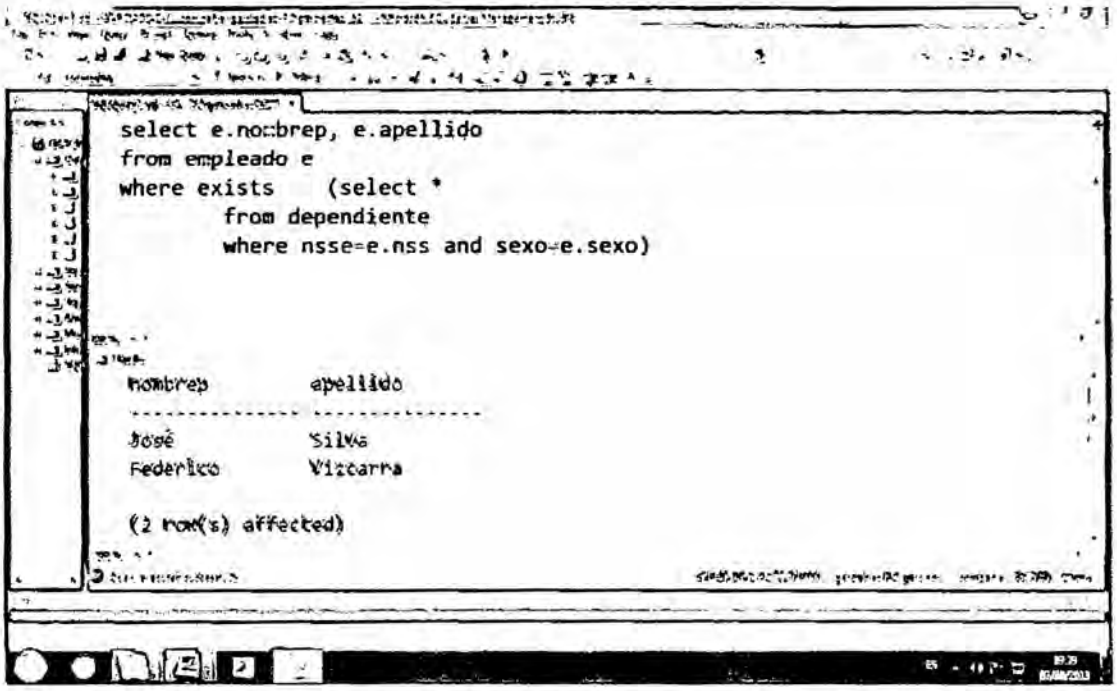

Figura 6.33 Comprueba si el resultado esta vacio. Fuente: Autor

Ejemplo: Listar los nombre de los Gerentes que tienen por lo menos un dependiente.

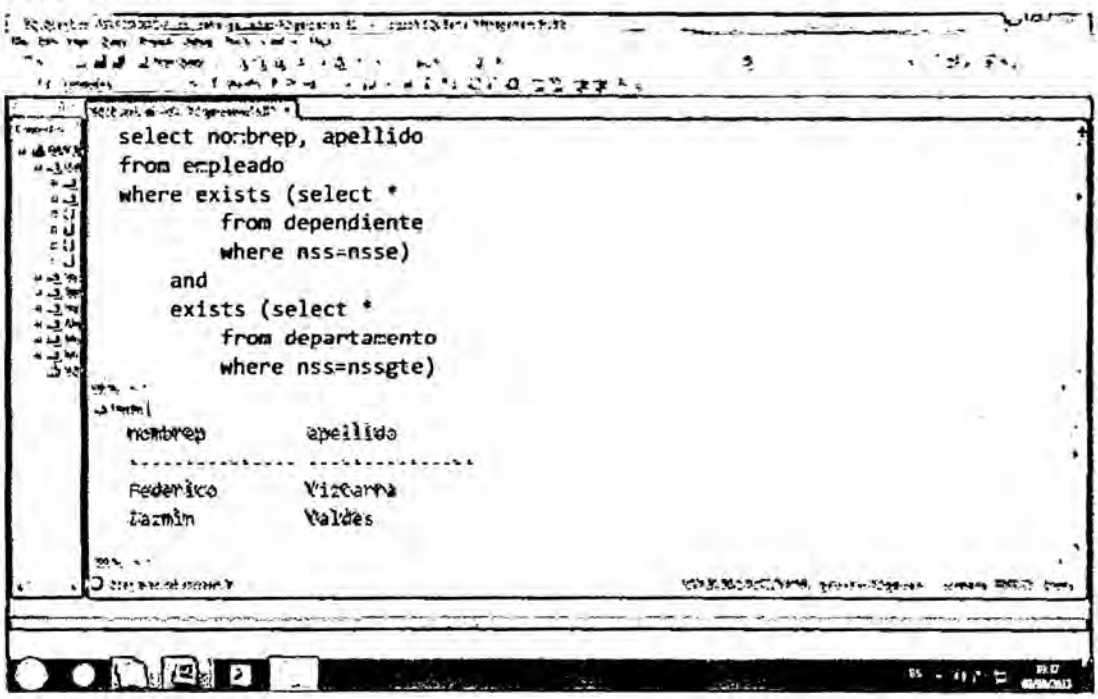

Figura 6.34 Comprueba si el resultado existe. Fuente: Autor

ORACLE

No existe el comando exists en Oracle.

Resultado:

La función exists sólo existe en SQL 2005, pero no existe en Oracle.

6.3.4.7 CONJUNTOS EXPLICITOS Y VALORES NULOS

SQL Server

Ejemplo: Obtener el numero de seguro social de todos los empleados que trabajan en los proyectos 1,2 o 3.

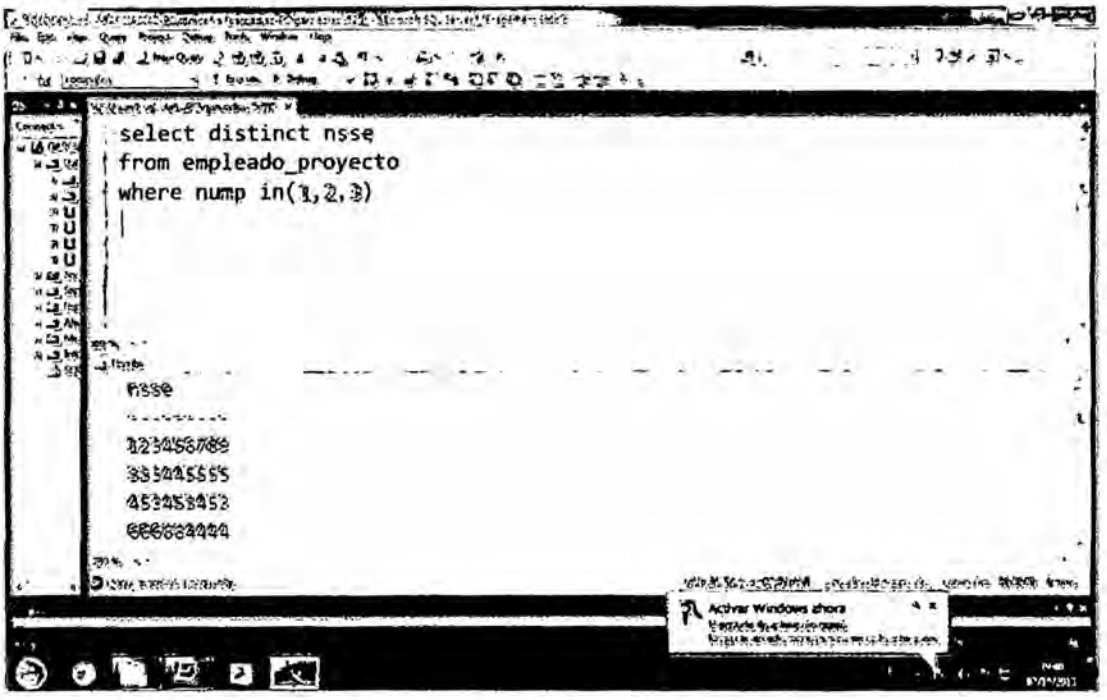

Figura 6.35 Utiliza un conjunto explicito de valores. Fuente: Autor

ORACLE

Ejemplo: Obtener los nombres de los empleados cuyo código de Jefe sea 100, 101 y 201.

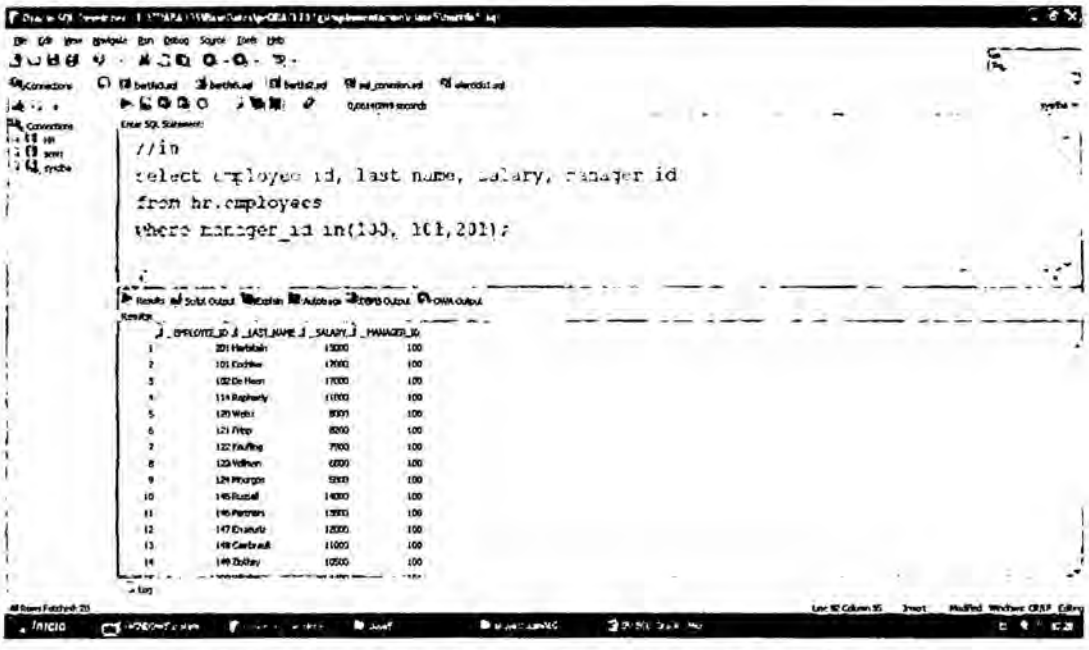

Figura 6.3.6 Utiliza un conjunto explicito de valores. Fuente: Autor

Ejemplo: Seleccionar los empleados cuyos jefes no tengan valores null.

Figura 6.37 Comprueba si un valor es null. Fuente: Autor

Resultado:

No hay diferencias en cuanto a conjuntos explicitos y valores nulos en ambos gestores de bases de datos.

6.3.4.8 CAMBIO DE NOMBRE DE LOS ATRIBUTOS

SQL Server

Ejemplo: Obtener el apellido de cada empleado y de su supervisor.

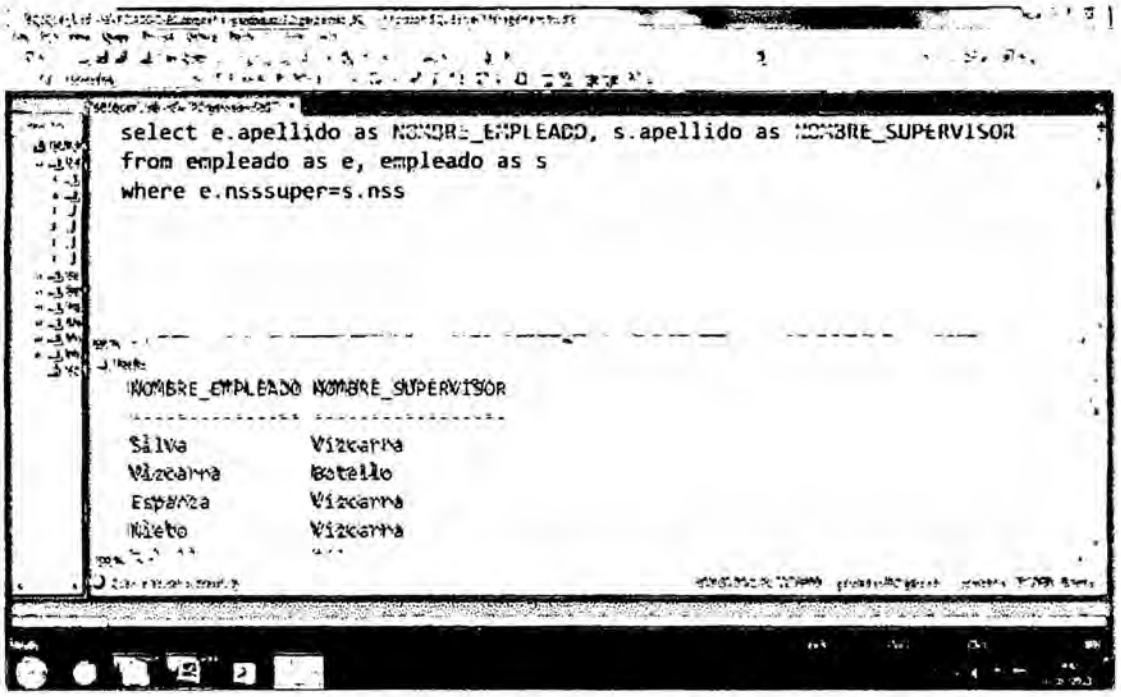

Figura 6.38 Consulta con cambio de nombre de los atributos. Fuente: Autor

ORACLE

Ejemplo: Obtener el apellido y salario de cada empleado.

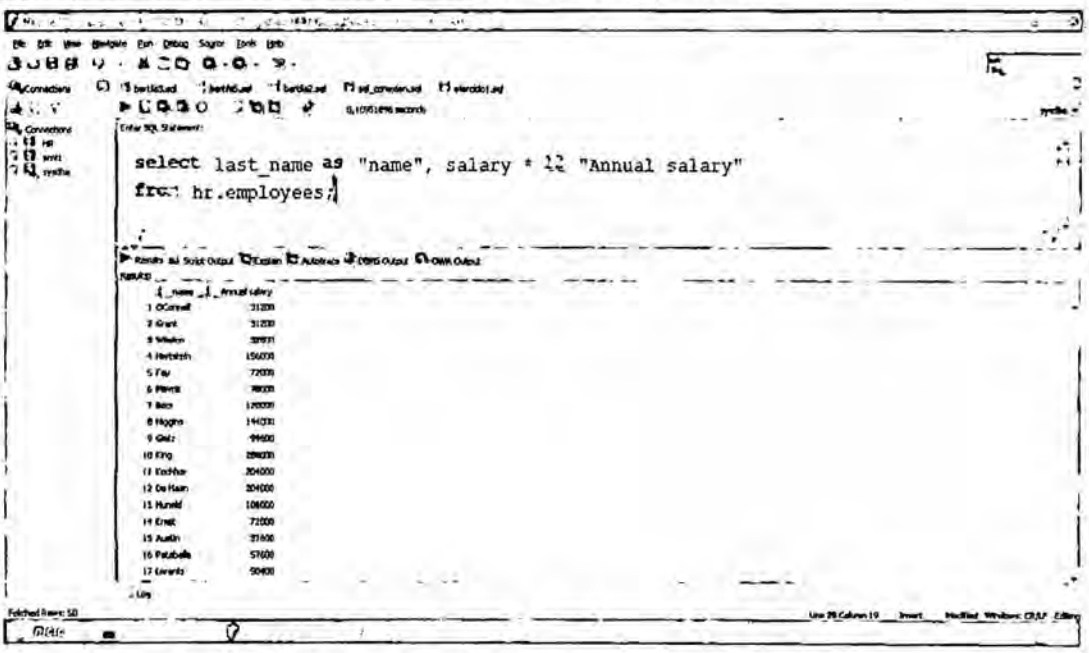

Figura 6.39 Cambio de nombre de los atributos. Fuente: Autor

Resultado:

No hay diferencias en cuanto a cambio de nombre de los atributos en ambos gestores de bases de datos.

6.3.4.9 FUNCIONES AGREGADAS Y AGRUPACIÓN

SQL Server

Ejemplo: Obtener la suma de los salarios; el salario máximo, mínimo y medio de todos los empleados del departamento de 'Investigación'.

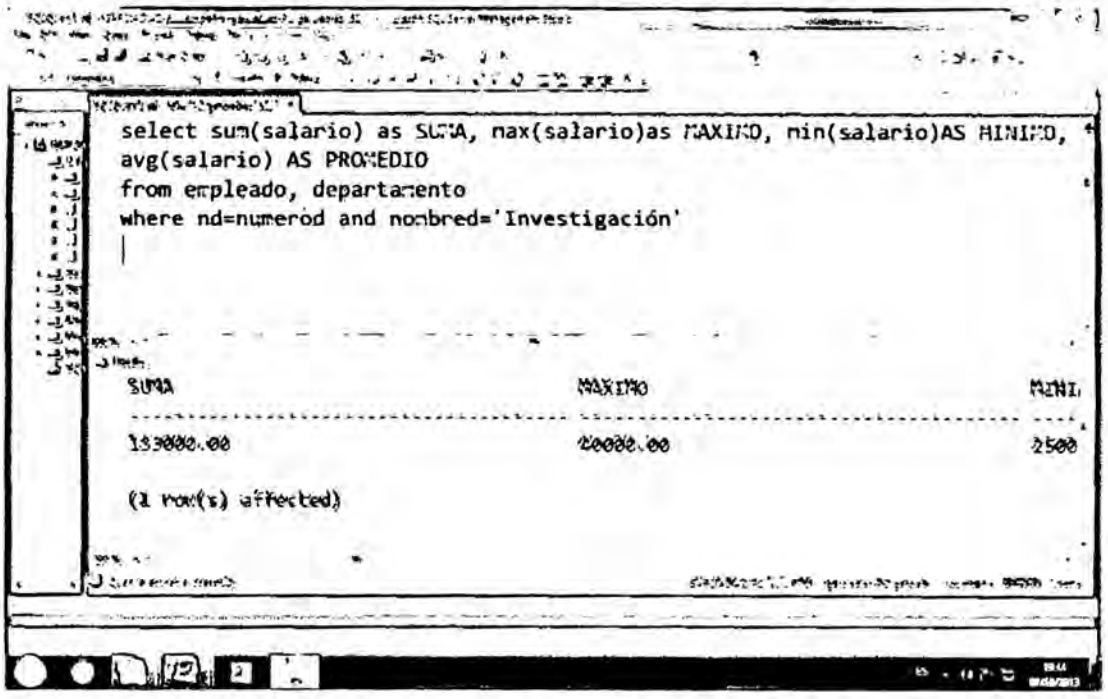

Figura 6.40 Uso de funciones agregadas. Fuente: Autor

Ejemplo: Para cada dpto, obtener el nro de dpto, nro de empleados del dpto y su salario medio.

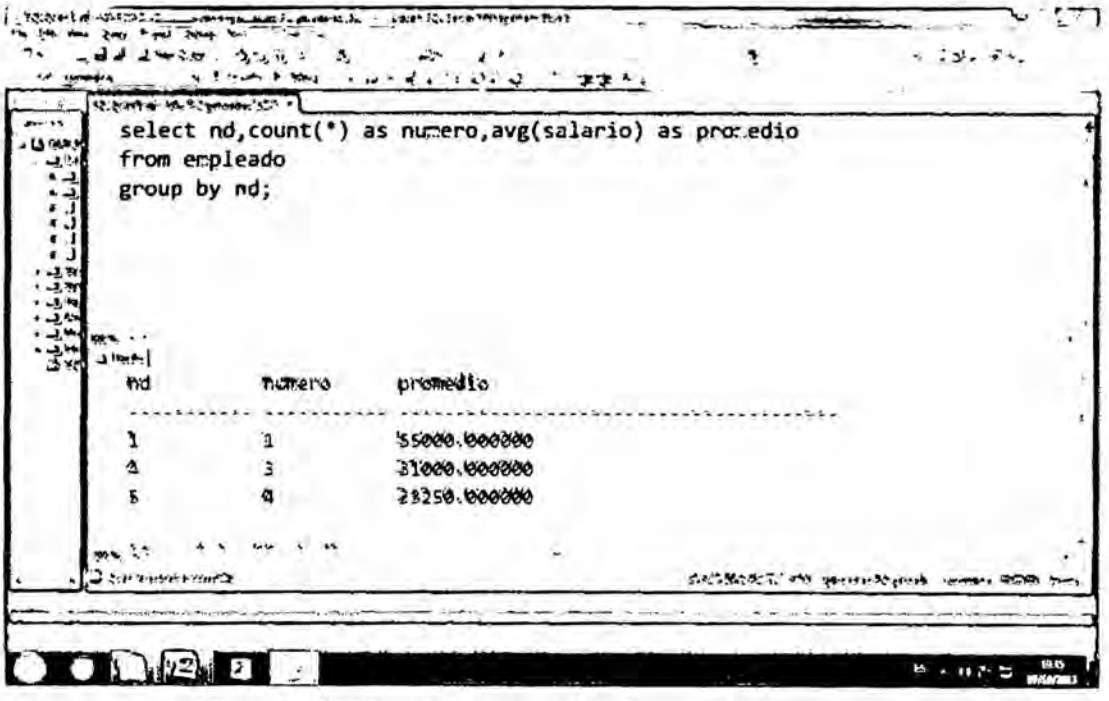

Figura 6.41 Uso de Agrupación. Fuente: Autor

Ejemplo: Para cada proyecto en el que trabajan más de 2 empleados, obtener el numero y el nombre del proyecto, así como el numero de empleados que trabajan en el.

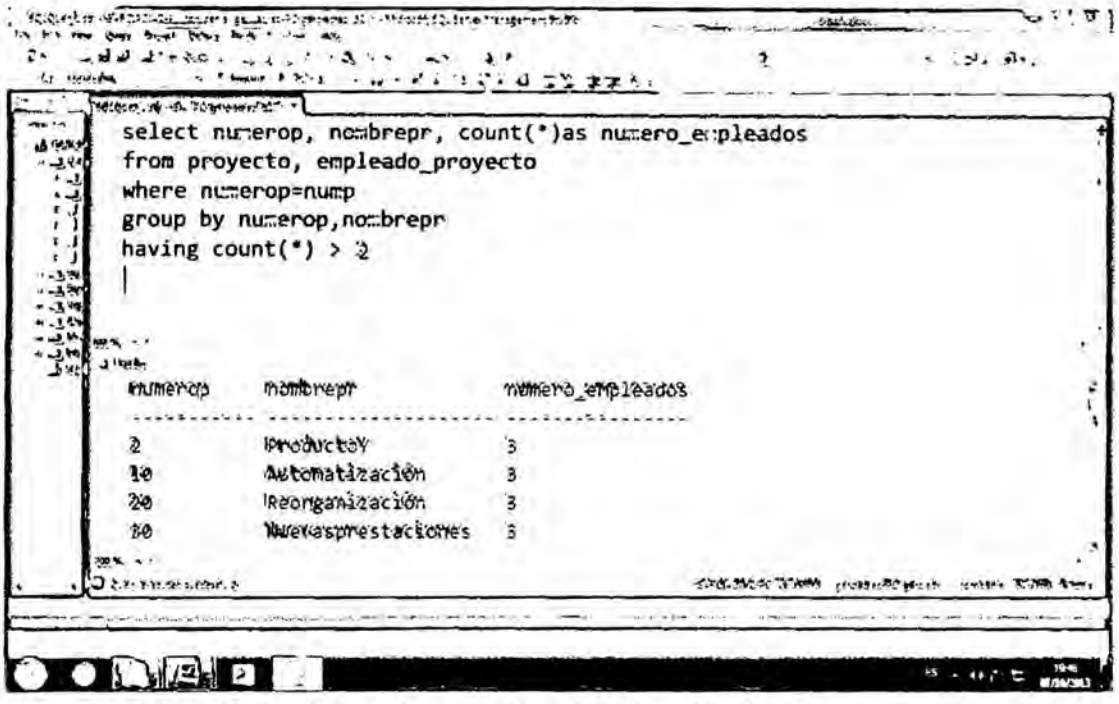

Figura 6.42 Uso de Agrupación y having. Fuente: Autor

ORACLE

Ejemplo: Obtener el salario medio, máximo, mínimo y total de los empleados cuyo job_id se parezca a REP.'

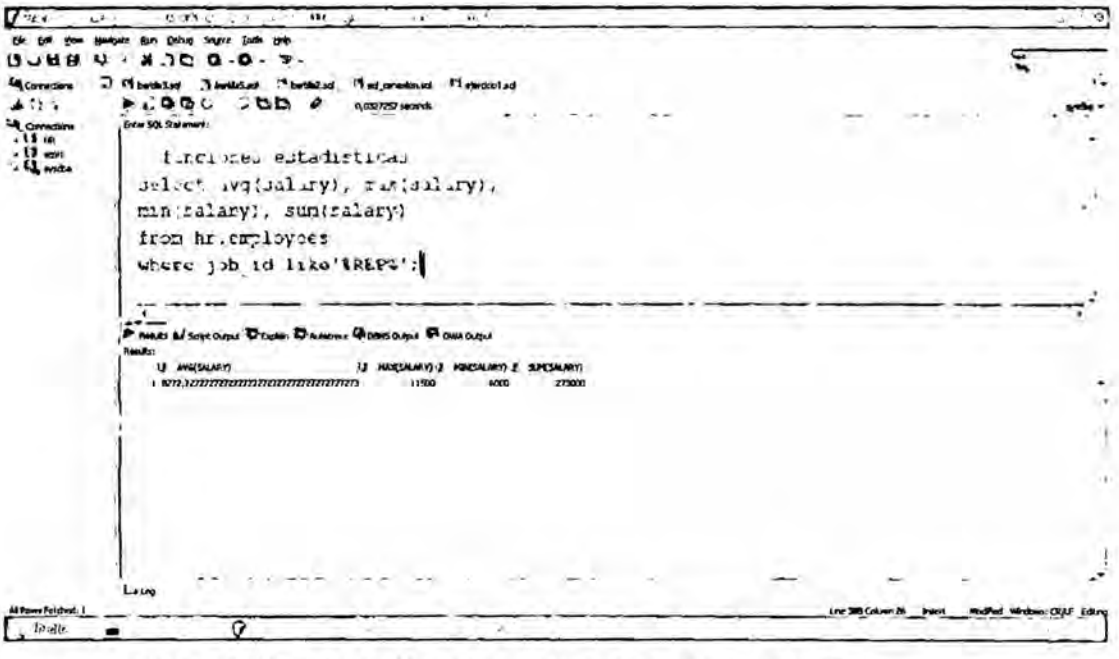

Figura 6.43 Uso de funciones agregadas. Fuente: Autor

Ejemplo: Mostrar el promedio de sueldo de cada Departamento.

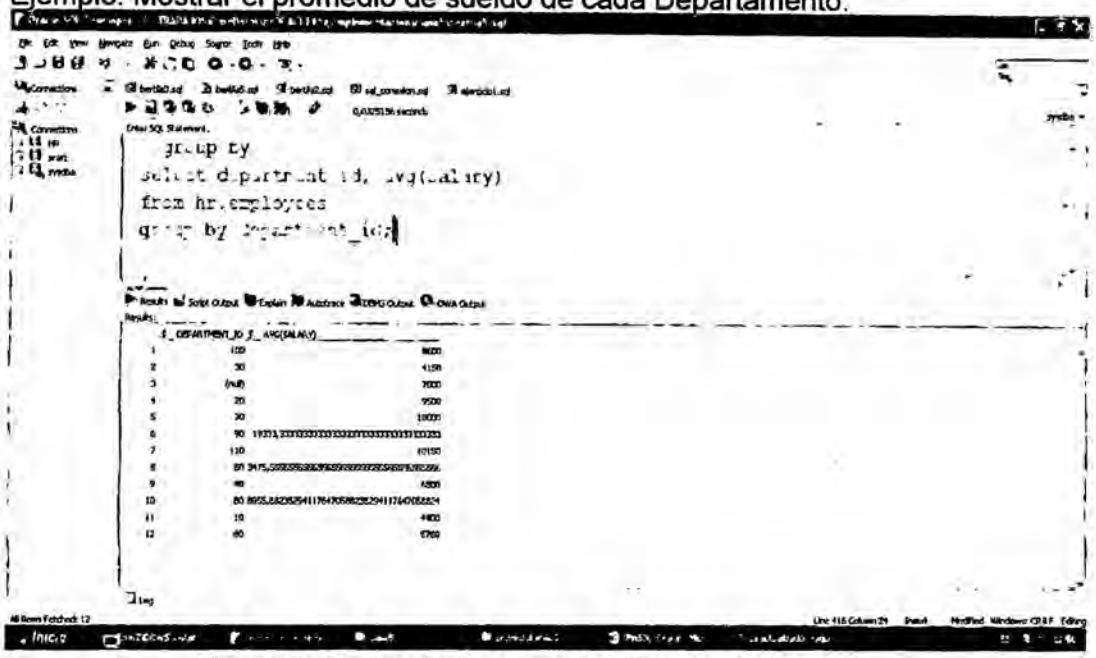

Figura 6.44 Uso de Agrupación. Fuente: Autor

Ejemplo: Mostrar el sueldo máximo de aquellos Departamentos, cuyos sueldos máximos sean mayores a 10000.

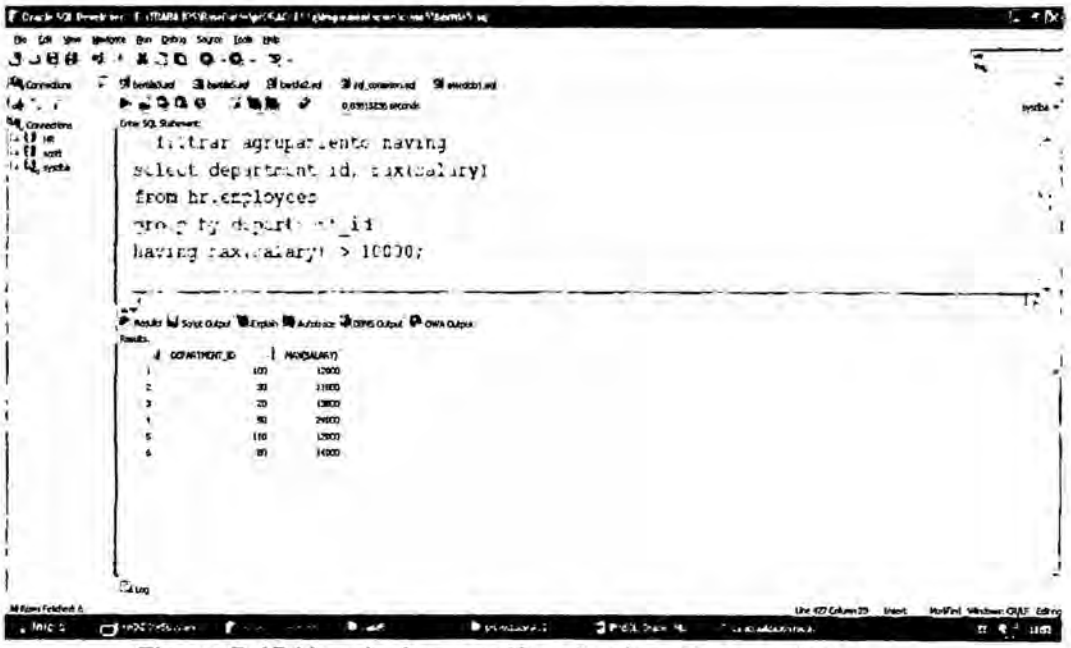

Figura 5.45 Uso de Agrupación y having. Fuente: Autor

Resultado:

No hay diferencias en cuanto a funciones agregadas y de agrupación en ambos gestores de bases de datos.

6.3.4.10 COMPARACIONES DE SUBCADENAS, OPERADORES ARITMÉTICOS Y ORDENACIÓN

SQL Server

Ejemplo: Obtener todos los empleados cuya dirección esté en Higueras, estado de México.

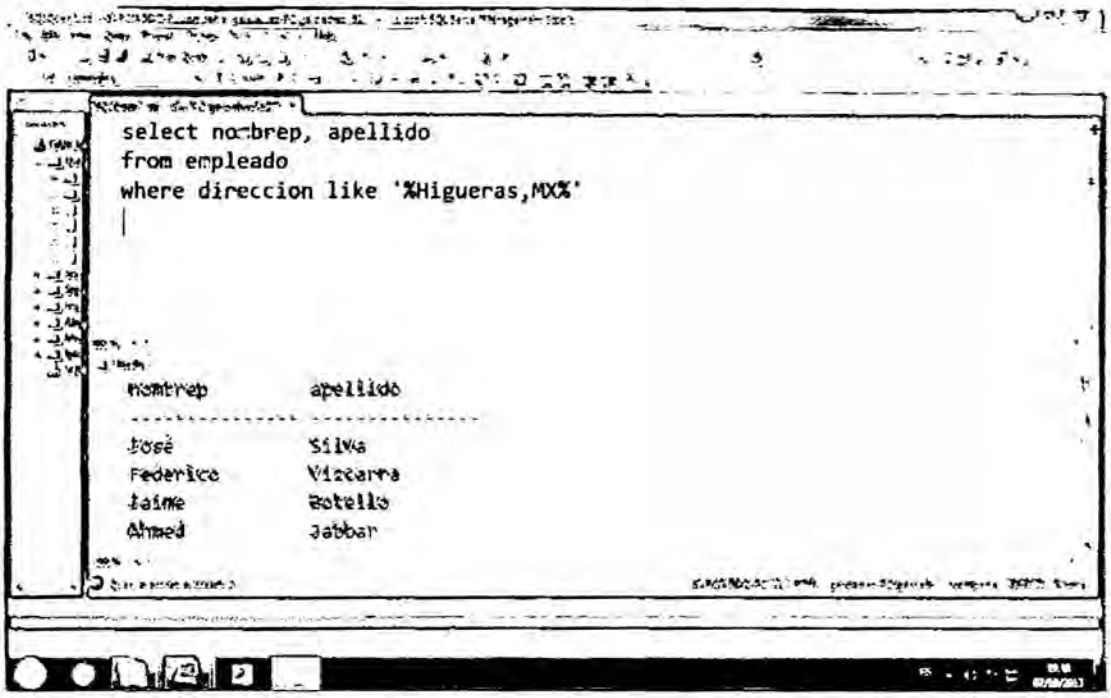

Figura 6.46 Uso de like. Fuente: Autor

Ejemplo: Mostrar los salarios resultantes si cada empleado que trabaja en el proyecto 'Producto X' recibe un aumento del 10%.

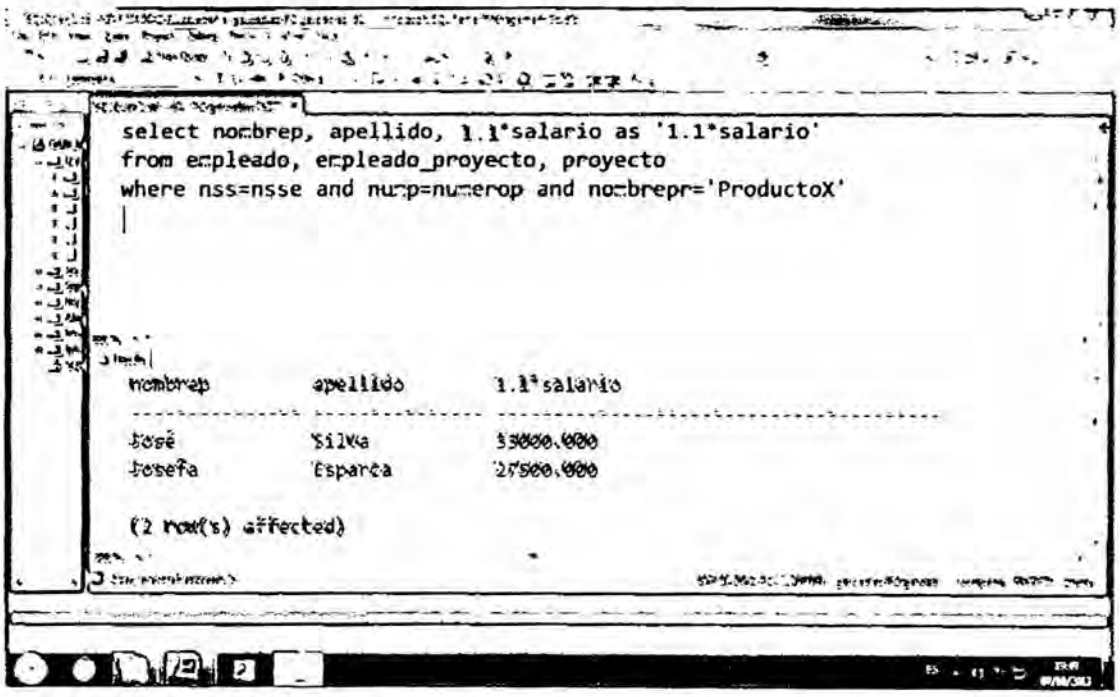

Figura 6.47 Uso de operadores aritméticos. Fuente: Autor

Ejemplo: Obtener una lista de empleados y de los proyectos en los que trabajan, ordenados por dpto y, dentro de cada dpto, alfabeticamente por apellido y nombre.

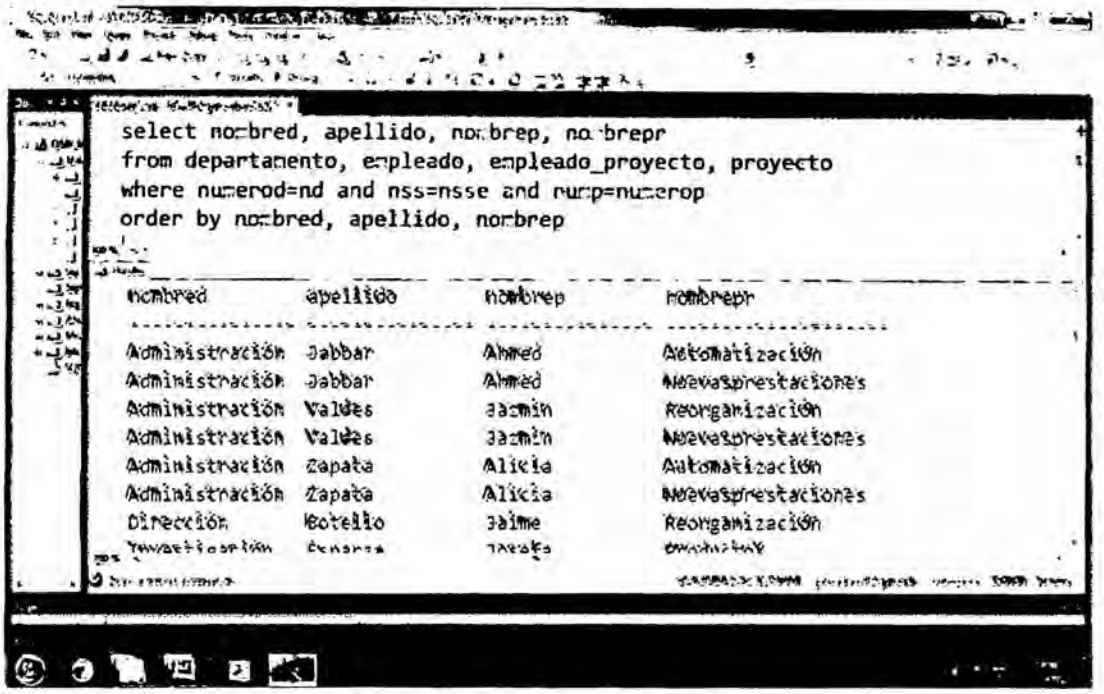

Figura 6.48 Uso de Order by. Fuente: Autor

ORACLE

Ejemplo: Obtener todos los empleados cuyo salario sea >= 10000 y su job_id sea parecido a MAN.

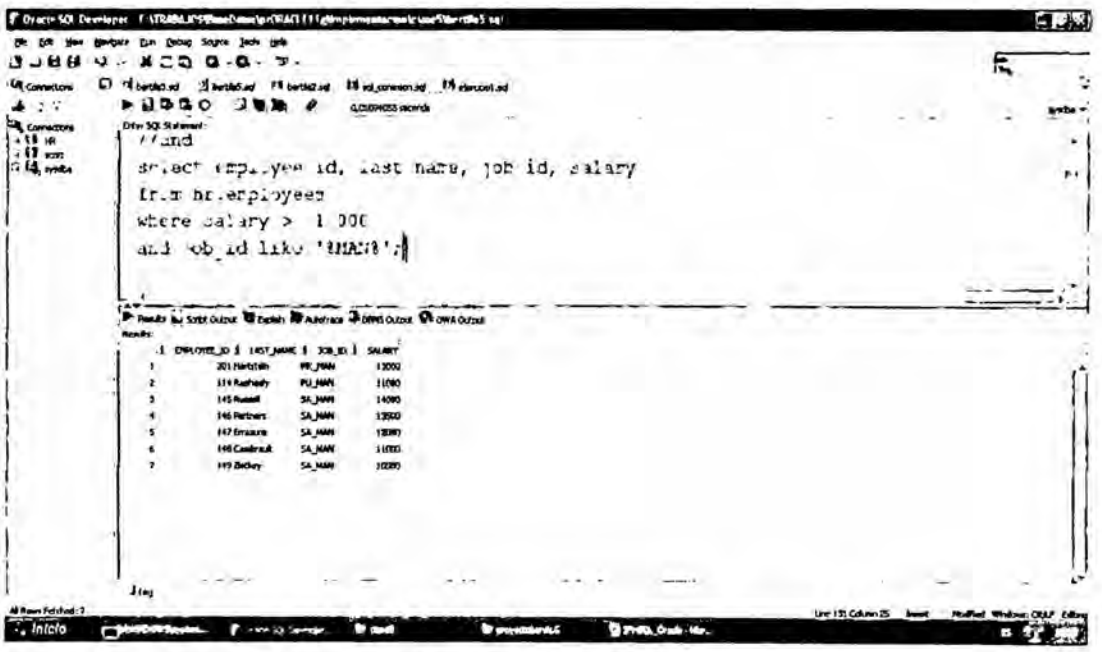

Figura 6.49 Uso de like. Fuente: Autor

Ejemplo: Mostrar los salarios resultantes si cada empleado que trabaja se le aumenta de acuerdo a una fórmula aritmética.

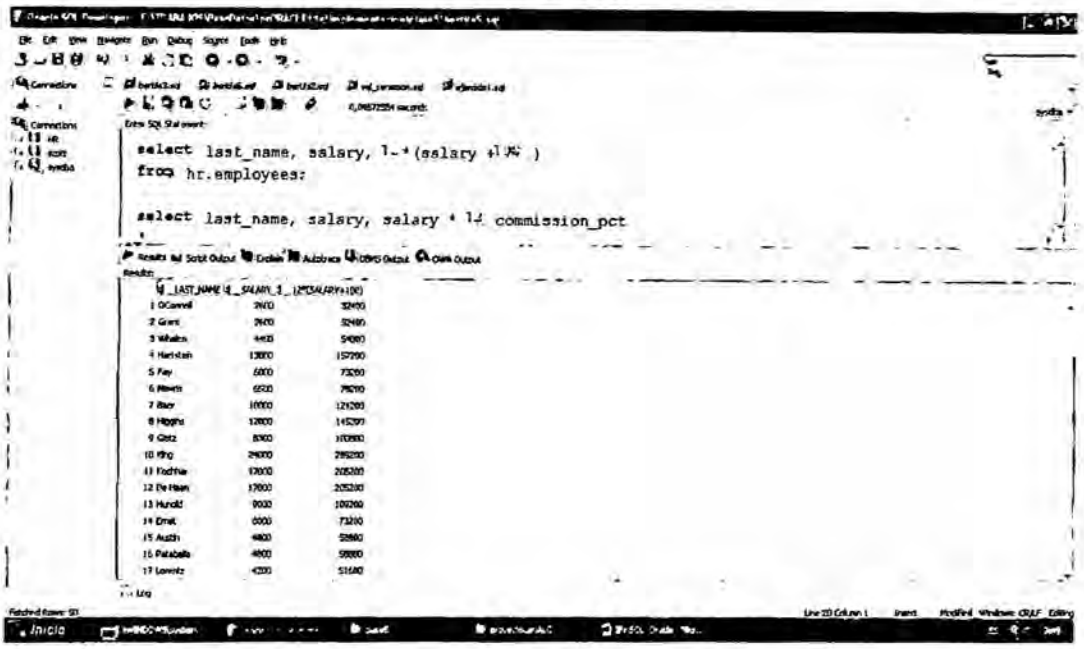

Figura 6.50 Uso de operadores aritméticos. Fuente: Autor

Ejemplo: Ordenar a los empleados por Departamento y dento de Departamento en forma descendente por sueldo.

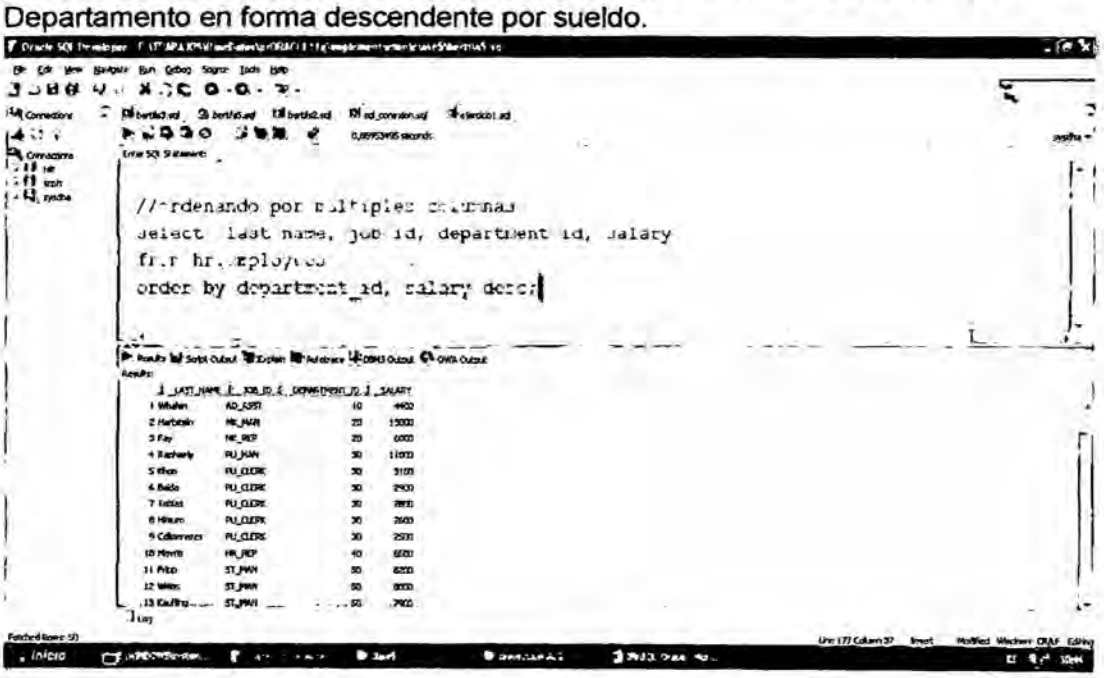

Figura 6.51 Uso de Order by. Fuente: Autor

Resultado:

No hay diferencias en cuanto a comparaciones de subcadenas, operadores aritméticos y ordenación en ambos gestores de bases de datos.

6.4 LENGUAJE DE CONTROL DE DATOS (DCL)

SQL Server

Ejemplo: Negar el acceso de inserción y eliminación de registros sobre las tablas pedidos y clientes.

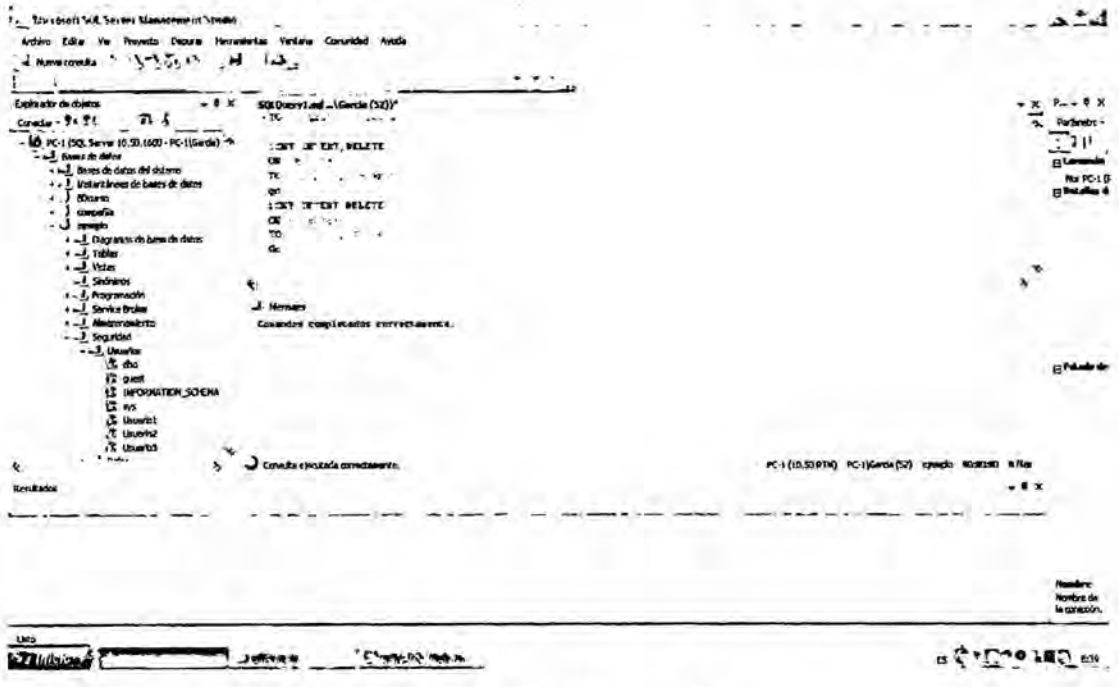

Figura 6.52 Uso de DENY. Fuente: Autor

Ejemplo: Al mostrar la Base de Datos en modo interactivo, se puede verificar los permisos y denegaciones para dicho usuario.

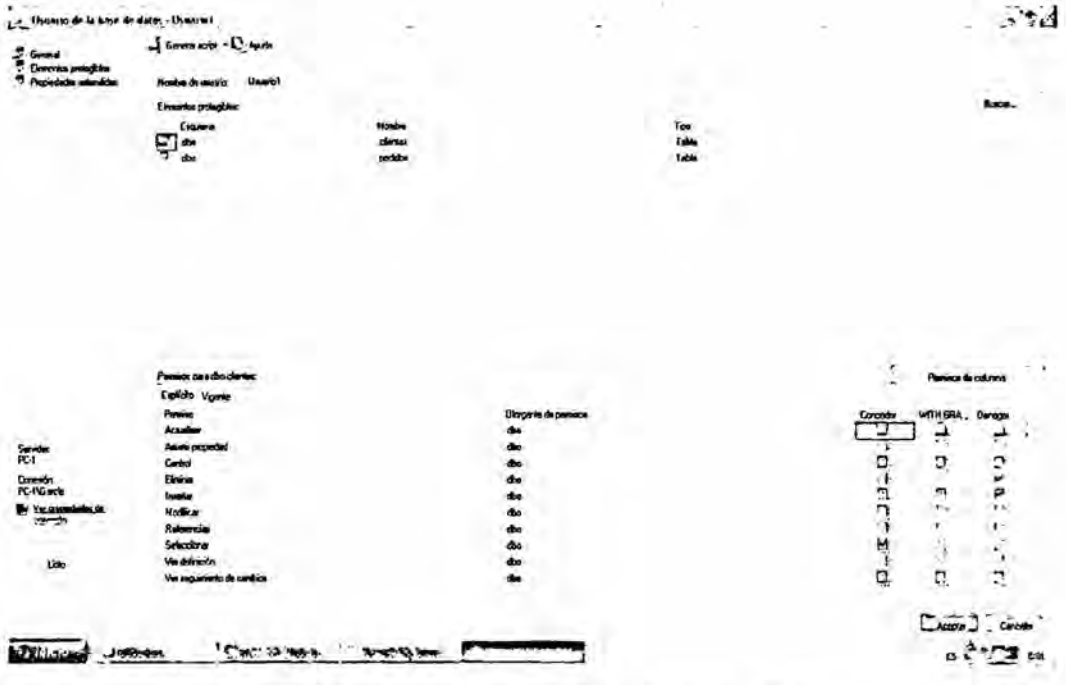

Figura 6.53 Verificación de DENY. Fuente: Autor

ORACLE

Ejemplo: El usuario Scott necesita consultar la tabla curso.

```
SQL> Grant Select On Curso to Scott;
Grant succeeded.
```

Ejemplo: Ahora hagamos la pueba respectiva.

```
SQL> connect Scott/tiger
Connected.

SQL> select * from egcc.curso;

IDCU NOMCURSO        VACANTES MATRICULADOS        PROFESOR        PRECURSO
C001 Oracle 11g      20             10            Gustavo Perez   350
```

Resultado:

Hay diferencias en la forma de otorgar los privilegios y autorizar el uso de los objetos en ambos gestores de bases de datos.

6.5 CUADRO COMPARATIVO RESUMEN DE LAS CARACTERISTICAS DE LOS LENGUAJES DE PROGRAMACIÓN ALGEBRAICOS

CARACTERISTICAS	SQL	ORACLE
Versiones y ediciones actuales (2013)	SQL server 2012	ORACLE 11 g R2
DDL (creación de BD, definición de tablas, etc)	X	X
DML (select, funciones de grupo, subconsultas..etc)	X	X
DCL (Administración de usuarios, creación y administración de roles, otorgar y quitar privilegios, etc)	X	X
Bloques, cursores y excepciones	X	X
Soporte de transacciones	X	X
Soporte de triggers	X	X
Soporta procedimientos almacenados	X	X
Soporte de Funciones	X	X
Modo Cliente - Servidor	X	X
Soporte OLAP y Minería de Datos	En proceso	100%
Soporte de SO	Solo Microsoft (XP, vista, windows server 2000, 2003 y 2008)	Multiplataforma (Windows, Linux, unix, etc)

Figura 6.54 Cuadro comparativo resumen de las caracteristicas de los lenguajes de programación algebraicos. Fuente: Autor

6.6 CONTRASTACION DE RESULTADOS ENTRE ORACLE Y SQL SERVER

	CONTRASTACION DE RESULTADOS ENTRE ORACLE Y SQL SERVER	
ITEM	SQL	ORACLE
SISTEMAS OPERATIVOS	XP, VISTA, Windows Server 2000, 2003 y 208, Windows (32 y 64 bits), Windows 7, Windows 8	Windows (32 y 64 bits), Linux, Unix (Solaris, HP_UX, AIX, etc)
VERSIONES	SQL 2008, SQL Server 2008R2, SQL 2012	10gR2, 11g, 11gR2
EDICIONES	En términos de ediciones de SQL Server 2008 R2 ofrece actualmente los siguientes: **Enterprise Edition**: La edición Enterprise tiene todas habilitadas las caracteristicas avanzadas y es apto para a gran escala, sitios de base de datos de alto volumen. **Standard Edition**: Este ofrece una plataforma asequible para las empresas que no requieren de las funciones avanzadas de la Enterprise Edition. La mayoría de las empresas suele desplegar sus bases de datos en las instancias de la edición estándar. **Workgroup Edition**: La edición de grupo de trabajo es adecuada para pequeñas aplicaciones departamentales e incorpora las características esenciales del producto. **Web Edition**: Esta es la destinada a ser utilizada por los proveedores de alojamiento como solución back-end de bajo coste para aplicaciones web. **Express Edition**: motor del servidor SQL embebido que puede ser utilizados para el almacenamiento local de datos y sistema de desarrollo de pequeñas escala. La edición Express se puede descargar gratis y pueden ser distribuida gratuitamente con un software. **Compact Edition**: La edición Compact permite a los usuarios desarrollar aplicaciones para computadoras de escritorio de Windows y dispositivos de mano.	Para Oracle 11g R2, los variantes son: **Enterprise Edition**: Esta ofrece el máximo rendimiento por tu dinero. Al igual que el SQLServer Enterprise Edition, todas las características y las capacidades del producto están permitido en esta edición. **Standard Edition**: Al igual que el estándar SQL Server Edition, la edición estándar de Oracle tiene habilitadas las principales características del producto y es adecuado para aplicaciones de negocios. **Standard Edition One**: Esta edición está diseñada para pequeños grupos de trabajo y la licencia para un máximo de 5 usuarios. **Express Edition**: En pequeña escala, base de datos inicial para fines de desarrollo y tiene licencia para redistribuirlo libremente. Express Edition 10g está todavía en la versión R2.

	Developer Edition: Todas las características de la versión Enterprise Edition está disponible en la edición para desarrolladores. Sin embargo, tiene licencia de uso por un usuario a la vez y está destinado a ser utilizado para fines de desarrollo y pruebas. Aparte de la versión Enterprise Edition, SQL Server 2008 R2 también ofrecerá dos ediciones "premium" para los grandes centros de datos y data warehouses. Estas ediciones se llamarán **Datacenter Edition** y el **Parallel Data Warehouse Edition**, respectivamente.	
COMPARACION ENTRE LAS DISTINTAS EDICIONES	Enterprise Edition Standard Edition Workgroup Edition Edition Express Web Edition Compact Edition Developer Edition	Enterprise Edition Standard Edition Standard Edition One Express Edition X X Enterprise Edition
NIVEL FÍSICO	bases de datos de sistema (**model**, **tempdb**, **master**, **msdb** y **resource**) y bases de datos de usuario. En el nivel físico, una base de datos de SQL Server está representada por un conjunto de archivos del sistema operativo que residen en el sistema de disco del servidor. Hay dos tipos de archivos de base de datos: el **archivo de datos** y el **archivo de registro de transacciones**.	Cuando Oracle se inicia, funciona igual que SQL en que una porción de la memoria del servidor se asigna para su funcionamiento. Esta área de memoria, conocido como el **Área Global de Sistema (SGA)**, se divide en una serie de estructuras diferentes. Junto con el espacio de memoria, una serie de procesos de fondo que también se inician para interactuar con el SGA. En conjunto, el espacio de memoria y los procesos constituyen una instancia de Oracle. Los archivos que componen una base de datos de Oracle se pueden clasificar en tres tipos: el **archivo de datos (data file)**, **archivo de rehacer (redo log file)** y el **archivo de control(control file)**. Los archivos de datos es donde residen todos los datos. Puede haber cualquier número de

		archivos de datos en una base de datos de Oracle. Archivos Rehacer son como los archivos de registro de transacciones de SQL Server que registra que cada cambio realizado a los datos y se utiliza para la recuperación del sistema. Los archivos de control son un tipo especial de archivo que contiene pequeñas piezas de información vital acerca de la base de datos. Sin este archivo, la instancia no será capaz de abrir la base de datos. Aparte de los archivos de datos, archivos rehacer y los archivos de control, la base de datos contendrá también un **archivo de parámetros**, y un **archivo de contraseñas**, opcionalmente, **archivos de registro de archivado** (archive log file). **Espacios de tablas (tablespaces).** Un espacio de tablas de Oracle es una estructura lógica que agrupa a las tablas, vistas, índices y otros objetos de la base de datos.

Figura 6.55 Contrastacion de resultados entre ORACLE y SQL server. Fuente: Autor

VII.- DISCUSIÓN

1.- La comparación de los resultados se realizará con los resultados de otros autores en los procedimientos a nivel del Lenguaje de definición de datos (DDL), Lenguaje de Modificación de Datos (DML) y Lenguaje de Control de Datos (DCL), que son los aspectos en que esta investigación ha realizado la comparación a nivel práctico de SQL y ORACLE.

2.- De acuerdo a la investigación realizada se puede concluir que a a nivel de Lenguajes de Definición de Datos (DDL) existen diferencias en cuanto a la creación de Base de Datos, en SQLserver se asigna la ubicación física de la Base de Datos y crea un archivo con extensión mdf y log. Y en Oracle se crea un Usuario y se le asigna en algún tablespace y luego se asignan privilegios para crear tablas.

No existen diferencias en el comando check constraint, en ambos SGBD.

Hay diferencias en la sintáxis para otorgar un valor por defecto en ambos Gestores de Bases de Datos ya que en Oracle es necesario ingresar el comando Modify.

Existen diferencias en la sintáxis para otorgar un valor por defecto en ambos Gestores de Bases de Datos ya que en Oracle es necesario ingresar el comando Modify.

Hay diferencias en los comandos para modificación y eliminación de columnas en ambos Gestores de Bases de Datos ya que en Oracle es necesario ingresar el comando Modify, pero no hay diferencias para adicionar columnas.

No hay diferencias en el comando Insert en ambos lenguajes algebraicos.

No hay diferencias en el comando Update para actualizar filas en ambos lenguajes algebraicos.

Hay una pequeña diferencia de sintaxis en el comando Delete, ya que en SQL no requiere de la clausula from.

3.- En cuanto al Lenguaje de Manipulación de Datos (DML) se comprobó las siguientes similitudes y diferencias:

No hay diferencias en cuanto a Sentencias Básicas en ambos SGBD.

No hay diferencias en cuanto al manejo de nombres de atributos ambiguos en ambos gestores de bases de datos.

No hay diferencias en cuanto al manejo de nombres de atributos ambiguos en ambos gestores de bases de datos.
No existen diferencias en cuanto al manejo de clausulas where no especificadas y empleo de *, en ambos gestores de bases de datos.
No hay diferencias en cuanto a tablas como conjuntos en ambos lenguajes algebraicos.
No existen diferencias en cuanto a consultas anidadas y comparaciones de conjuntos en ambos gestores de bases de datos.
La función exists sólo existe en SQL 2005, pero no existe en ORACLE.
No hay diferencias en cuanto a conjuntos explicitos y valores nulos en ambos gestores de bases de datos.
No existen diferencias en cuanto a cambio de nombre de los atributos en ambos gestores de bases de datos.
No hay diferencias en cuanto a funciones agregadas y de agrupación en ambos gestores de bases de datos.
No existen diferencias en cuanto a comparaciones de subcadenas, operadores aritméticos y ordenación en ambos gestores de bases de datos.

4.- En lo referente a Lenguaje de Control de Datos, se halló las siguientes similitudes y diferencias:
Existen diferencias en la forma de otorgar los privilegios y autorizar el uso de los objetos en ambos gestores de bases de datos.

5.- Podemos concluir que las diferencias entre ambos lenguajes es mayor a nivel de Lenguaje de definición de datos (DDL), siendo mínima la diferencia en el Lenguaje de Manipulación de Datos (DML) y existen pequeñas diferencias a nivel del Lenguaje de Control de Datos (DCL).

6.- También podemos concluir que existen diferencias a nivel de los sistemas operativos donde corren estos SGBD, mientras que ORACLE es multiplataforma, SQL server sólo se ejecuta en el Sistema Operativo de Windows de Microsoft.
También existe diferencias en cuanto a costos, ya que el costo de utilizar el ORACLE es mayor que en SQL server.
Desde el punto de vista del software, ORACLE ofrece más ventajas, como por ejemplo el uso de DATAWAREHOUSE y MINERIA DE DATOS, que

forman parte del Lenguaje. SQL también busca mejorar estas características. Ambos utilizan el Modelo Relacional.

7.- Se podría enseñar SQL server en un curso inicial de Base de Datos, por su mayor simplicidad y en un curso más avanzado ORACLE, por ser más complejo.

VIII.- REFERENCIALES

1.- DE MIGUEL, ADORACIÓN Y PIATTINI, MARIO. Fundamentos y Modelos de Bases de Datos, México: Editorial Alfaomega, 2da edición, 1999.

2.- ELMASRI/ NAVATHE. Fundamentos de Sistemas de Bases de Datos, México: Editorial Prentice Hall, 5ta edición, 2007.

3.- REINOSA, ENRIQUE JOSÉ / MALDONADO, CALIXTO ALEJANDRO,..OTROS. Base de Datos, Argentina: Editorial Alfaomega Grupo editor Argentino, 1era edición, 2012.

4.- SILBERSCHATZ, ABRAHAM Y KORTH, HENR, Fundamentos de Bases de Datos. Madrid: Editorial Mc Graw Hill, 4ta Edición. 2002.

BIBLIOGRAFÍA VIRTUAL

* *Documentos de Internet, relacionados al tema*

1.- http://www.uhu.es/jacinto_mata /*Universidad de Huelva*/

IX.- APÉNDICE

ADICIONALES DE SQL

1.- /* ex1 obtener nombre, apellido y nombre del departamento de los empleados nacidos entre 1950 y 1960*/

```
select nombrep, apellido, nombred
from empleado, departamento
where (nd=numerod) and fechan between '19550101' and '19601231'
```

nombrep	apellido	nombred
José	Silva	Investigación
Ahmed	Jabbar	Administración
Alicia	Zapata	Administración

2.- /* ex2 Para cada departamento obtener el nombre del departamento, así como el numero de personas que trabajan en el, ordenado alfabéticamente*/

```
select nombred as Departamento, count(*) as NroPersona
from empleado, departamento
where (nd=numerod)
group by nombred
order by nombred
```

Departamento	NroPersona
Administració	3
Dirección	1
Investigación	4

3.- /* ex3 Para cada departamento , donde trabajan más de 3 personas, obtener el nombre del departamento, así como el número de personas que trabajan en el, ordenado alfabéticamente*/

```
select nombred as departamento, count(*) as NroPersonas
from empleado, departamento
where (nd=numerod)
group by nombred
having count(*) > 3
order by nombred
```

Departamento	NroPersonas
Investigación	4

4.- /* Mostrar nombre, apellido, salario de los empleados que trabajan en el departamento de investigación si tuvieran un aumento del 25%*/

```
select nombrep, apellido ,1.25*salario as NuevoSalario
from empleado, departamento
where nd=numerod and nombred='Investigación'
```

```
nombrep  apellido     NuevoSalario
José     Silva       37500.0000
Federico Vizcarra    50000.0000
Josefa   Esparz      31250.0000
Ramón    Nieto       47500.0000
```

5.- /*ex5 prepare una lista de nombre y apellidos de todos los gerentes de departamento que tienen hijos con su mismo sexo*/

```
select distinct nombrep, apellido
from empleado, dependiente, departamento
where nss=nsse and nd=numerod and empleado.sexo= dependiente.sexo
and (parentesco='hija'  or parentesco='hijo')
```

```
nombre   apellido
Federico Vizcarra
José     Silva
```

6.- /*ex6-obtener el número de empleados que trabajan en el departamento ='investigación'*/

```
select count(*) as NroEmpleadosInvestigación
from empleado
where nd in (select numerod
             from departamento
             where  nombred='investigación')
```

```
NroEmpleadosInvestigación
4
```

7.- /*ex7 obtener el número de empleados que trabajan en el departamento de 'investigación'*/

```
select count(*) as NroEmpleadosInvestigación
from empleado, departamento
where nd=numerod and  nombred='investigación'
```

```
NroEmpleadosInvestigación
4
```

8.- /*cc16 Para cada proyecto obtener el número y el nombre del proyecto, así como el promedio de sueldos de los empleados que trabajan en él.*/

```
select numerop, nombrepr, avg(salario) as SalarioProm
from proyecto, empleado_proyecto, empleado
where numerop=nump and nsse=nss
group by numerop, nombrepr
```

numerop	nombrepr	SalarioProm
1	ProductoX	27500.000000
2	ProductoY	31666.666666
3	ProductoZ	39000.000000
10	Automatizació	30000.000000
20	Reorganización	46000.000000
30	Nuevasprestaciones	31000.000000

9.- /*cc17 Mostrar los salarios resultantes si cada empleado que trabaja en el proyecto "Reorganización" recibe un descuento del 10%.*/

```
select nombrep, apellido, 0.9*salario as Descuento
from empleado, empleado_proyecto, proyecto
where nss=nsse and numerop= nump and nombrepr='Reorganización'
```

nombrep	apellido	Descuento
Federico	Vizcarra	36000.000
Jaim	Botello	49500.000
Jazmin	Valdes	38700.000

10.- /*cc18 Obtenga los nombres de todos los empleados del departamento 4 que trabajan más de 20 horas por semana en el proyecto 'Automatización'*/

```
select nombrep, apellido, sum(horas)as horas
from empleado, empleado_proyecto
where nd=4 and nss=nsse
and nump in (select numerop
        from proyecto
        where nump =numerop and nombrepr = 'Automatización')
group by nombrep, apellido
having sum(horas) > 20
```

nombrep	apellido	horas
Ahmed	Jabbar	35.0

11.- /*cc19 Obtener el nombre de todos los empleados que tienen cónyuge.*/

```
select e.nombrep, e.apellido
from empleado e
where e.nss in (select nsse
        from dependiente
        where nsse=e.nss and parentesco='cónyuge')
```

nombrep	apellido
José	Silva
Federico	Vizcarra
Jazmin	Valdes

12.- /*cc20 Seleccionar nombre y apellido de los Empleados que nacieron entre el 58 y 60.*/

```
select nombrep, apellido
from empleado
where fechan between '01/01/58' and '31/12/60'
```

```
nombrep  apellido
Ahmed    Jabbar
Alicia   Zapata
```

UNIVERSIDAD NACIONAL DEL CALLAO
Facultad de Ingeniería Industrial y de Sistemas
Escuela Profesional de Ingeniería de Sistemas

SYLLABUS

1.- INFORMACION GENERAL

1.1 Nombre de la Asignatura	: Base de Datos
Nro	: 43
Código del Curso	: PCO83
1.2 Carácter	: Obligatorio
1.3 Pre-requisito	: Lenguaje de Programación III
1.4 Número de créditos	: 04
Horas semanales	: Seís(06):
Teoría	: 02 Hrs.
Práctica	: 02 Hrs.
Laboratorio	: 02 Hrs.
1.5 Ciclo Académico	: Octavo Ciclo
1.6 Semestre Académico	: 2013-II
1.7 Duración	: 17 semanas
1.8 Profesora	: Mg Bertila García Díaz

2.- SUMILLA

Proporcionar al estudiante los elementos básicos del procesamiento de datos. Modelamiento de BD. Modelo Relacional, Álgebra Relacional. SQL. Normalización. Datawarehouse y Bases de Datos distribuidas.

3.- OBJETIVO GENERAL:

Conocer las técnicas de Diseño que permitan diseñar una Base de Datos en todos sus aspectos.

4.- COMPETENCIAS

- Domina una Metodología de Diseño de Base de Datos.
- Diseña Bases de Datos usando el modelo relacional.
- Crea Bases de Datos usando el SGBD SQL, realiza consultas, passwords. Etc.
- Diseña Bases de Datos a partir de Documentos.
- Comprende la importancia del Datawarehouse en la toma de decisiones.

5.- PROGRAMACIÓN DE LOS CONTENIDOS

UNIDAD I: ARQUITECTURA PARA SISTEMAS DE BASES DE DATOS

Contenidos Procedimentales

- Describe los diferentes niveles de la arquitectura de las Bases de Datos.

Contenidos Actitudinales

- Distingue los diferentes niveles de la arquitectura de las Bases de Datos.

Contenidos Conceptuales

Primera Semana
Introducción a los Sistemas de Bases de Datos. Diferencia entre archivos y Bases de Datos. Definición y Arquitectura de un Sistema de Bases de Datos.

UNIDAD II: MODELADO DE DATOS CON EL ENFOQUE ENTIDAD-RELACION

Contenidos Procedimentales
- Esboza modelos entidad- relación

Contenidos Actitudinales
- Expone diferentes ejemplos de modelos entidad- relación

Contenidos Conceptuales

Segunda Semana
Enfoque de modelo de Datos Entidad-Relación, Guías para determinación de Entidades, Atributos, Relacionamientos. **Laboratorio**: Pr. de Modelamiento Nº 1

Tercera Semana
Asociaciones básicas del Modelamiento de datos. Controles de Negocios: opcional, mandatoria. **Laboratorio**: Pr. de Modelamiento Nº 2 - Pr. Nº 3 de Erwin.

Cuarta Semana
Extensiones al modelo E-R: Clasificación, Generalización, Agregación. **Laboratorio**: Pr. de Modelamiento Nº 2 - Pr. Nº 4 de Ing. Delantera y Reversa.

UNIDAD III: MODELO RELACIONAL Y ALGEBRA RELACIONAL

Contenidos Procedimentales
- Modifiça el Modelo Entidad – Relación al modelo Relacional

Contenidos Actitudinales
- Transforma el Modelo Entidad – Relación al modelo Relacional.

Contenidos Conceptuales

Quinta Semana
Modelo Relacional: Conceptos, relaciones, dominios, tuplas, claves primarias y foráneas, reglas de integridad. **Laboratorio**: Pr. Nº 5 de Modelo Relacional.

Sexta Semana
Modelo Relacional: Algebra Relacional. Operaciones con algebra relacional. **Laboratorio**: Pr. Nº 5b de Algebra Relacional. **1º Práctica calificada de ERWIN**

Setima Semana
Conversión del Modelo ER al Modelo Relacional. **Laboratorio**: Pr. Nº 5c de Conversión del Modelo ER al Modelo Relacional. **Exposición del Avance ER.**

Octava Semana
Examen Parcial.

UNIDAD IV: SQL

Contenidos Procedimentales
- Administra una Base de datos en un SGBD como SQL server

Contenidos Actitudinales
- Ejemplifica una Base de datos en un SGBD como SQL server.

Contenidos Conceptuales

Novena Semana
SQL: características, definición de tablas, reglas de Integridad, manipulación de datos y desarrollo de Aplicaciones. **Laboratorio**: Pr. Nº 6 Introducción al SQL.

Decima Semana
Sistema Relacional: Vistas, SQL embebido. DDL(data definition language). **Laboratorio**: Pr. Nº 7 SQL – DDL

Onceava Semana
SQL. DML (data manipulation language), DCL (data control language). Laboratorio con SQL SQL2008. **Laboratorio**: Pr. Nº 8 SQL – DDL - Pr. Nº 9 SQL – DML.

UNIDAD V: NORMALIZACION

Contenidos Procedimentales
- Esboza el modelo físico a partir de un documento.

Contenidos Actitudinales
- Diseña el modelo físico a partir de un documento.

Contenidos Conceptuales

Doceava Semana
Normalización: Primera, Segunda, Tercera y Cuarta Forma Normal. Aplicaciones. **Laboratorio**: Pr. Nº 10 SQL - DML. (Compañía) - Pr. Nº 12 Normalización.

Treceava Semana
Cuarta Formal Normal. Ejemplos de Aplicaciones. **Laboratorio**: Pr. Nº 11 SQL – DCL - Pr. Nº 12 Normalización
UNIDAD VI: DATAWAREHOUSE Y BASES DE DATOS DISTRIBUIDAS

Contenidos Procedimentales

- Esboza la importancia del Business Intelligence en la toma de decisiones.

Contenidos Actitudinales

- Comunica la importancia del Business Intelligence en la toma de decisiones.

Contenidos Conceptuales

Catorceava Semana
Fundamentos, Diseño y Construcción, Uso del Datawarehouse, Minería de Datos
Laboratorio: Pr. Nº 13 Procedimientos Almacenados - Pr. Nº 14 Funciones.

Quinceava Semana
Sistemas distribuidos; Bases de datos distribuidas: integridad, recuperación, concurrencia. Sistemas Cliente/Servidor. **2da Práctica calificada de SQL**

Dieciseisava Semana
Examen Final.

Diecisieteava Semana
Examen Sustitutorio.

6.- CRITERIOS DE EVALUACION:

N1: Nota de Primer parcial (Primera Parte)
N2: Una nota de Trabajos Prácticos
N3: Una nota de Examen Final
Promedio General = (N1 + N2 + N3)/3

7.- METODOLOGIA

La metodología empleada, será activa, que favorezca la participación del alumno, Se estimulará el interés por la investigación científica.

- Exposición sobre el avance del proyecto final
- Prácticas dirigidas y calificadas en el Laboratorio de Computación.

8.- BIBLIOGRAFIA

8.1 Bibliografia Basica

* Sistemas de Bases de Datos ***($$)***
Elmasri/ Navathe, Edi. Addison-Wesley Iberoamericana, S.A. 2º edición, 2000, 887 pg.

* Fundamentos de Sistemas de Bases de Datos
Elmasri/ Navathe, Edi. Prentice Hall, 5ta edición, 2011

* Base de Datos

Enrique José Reinosa/ Calixto Alejandro Maldonado..otros- - Edi. Alfaomega Grupo editor Argentino, 2012

8.2 Bibliografía Complementaria

* Introducción a los sistemas de Bases de Datos
Date C.J. Edi. Addison-Wesley Iberoamericana, 7ema edición, 2001, 936 pag.

* Bases de Datos desde Chen hasta Codd con Oracle,
Luque Ruiz, Irene- Gomez Nieto, Miguel Angel, Edi. AlfaOmega-Rama, 2002

* Diseño y Administración de Bases de Datos,
Gary W. Hansen, James V. Hansen, Edi. Prentice Hall, 2da edición, 2000

* Fundamentos y Modelos de Bases de Datos
Adoración de Miguel y Mario Piattini, Edi. Alfaomega, 1999, México.

* Microsoft SQL Server 2008 Implementación
Manual de Sistemas UNI

* *Fundamentos de Bases de Datos ($$)*
Abraham Silberschatz, Henry Korth, Edi Mc Graw Hill, 2002

X.- ANEXOS

L.- PERMISOS

1.- DAR PERMISOS

GRANT {SELECT ¦INSERT(campo) ¦DELETE ¦UPDATE(campos) ¦ REFERENCES(campo)}
ON objeto TO {usuario ¦PUBLIC} [WITH GRANT OPTION]

La opción WITH GRANT OPTION da la posibilidad de propagar los propios privilegios libremente.

2.- QUITAR PERMISOS

REVOKE [GRANT OPTION FOR] permisos ON objeto FROM usuarios
{RESTRICT ¦CASCADE}

La opción GRANT OPTION FOR da la posibilidad de propagar los propios privilegios libremente.

3.- VISTAS

CREATE VIEW nombre AS SELECT campos FROM tabla

La vista tiene los mismos permisos que la tabla (el creador tiene que tener al menos el permiso SELECT sobre la tabla).

- las listas, ya sean de valores, de campos o de tablas, se separan por comas.
- Lo indicado entre corchetes es opcional.
- Lo indicado entre llaves es una lista de opciones de la que debe escogerse una.

NOTA:

- Se indica en color rojo los signos de puntuación y similares necesarios para una correcta sintaxis.
- Se indica en mayúsculas, de color azul y subrayadas las palabras reservadas del lenguaje SQL.

J.- CONSULTAS DE ACCION

1.- DELETE

DELETE campos FROM tabla WHERE criterio ;

Para múltiples tablas, en "campos" debe especificarse <tabla>.<campo>
(todas deben ser de multiplicidad M:1).

Para todos los registros, en "campos" debe especificarse <tabla>.*

2.- INSERT

INSERT INTO tabla [campos] VALUES valores

Para insertar registros de otra tabla (tabla2):

INSERT INTO tabla [IN base_datos-externa] (campos)
SELECT tabla2.campoi, . . . , tabla2.campoj FROM tabla2 ;

Si queremos insertar registros de otra tabla (tabla2) que tiene igual estructura:

INSERT INTO tabla SELECT tabla2.* FROM tabla2 ;

3.- UPDATE

UPDATE tabla SET campo_1 = expresión_1, . . . , campo_n = expresión_n
[WHERE criterio] ;

K.- CLAÚSULAS DE UTILIDAD:

1.- EXIST - NOT EXIST

WHERE [NOT] EXISTS (subselect)

EXISTS devuelve cierto si la SUBSELECT devuelve al menos 1 valor.
NOT EXISTS devuelve cierto si la SUBSELECT no devuelve ningún valor.

2.- ORDER BY

SELECT campos FROM tabla WHERE . . . GROUP BY . . . HAVING . . .
ORDER BY {nombre_columna ¦ posición_columna} { ASC ¦ DESC }

Ordena una salida de consulta por un campo en modo ascendente (ASC) o descendente (DESC).

3.- UNION – UNION ALL

SELECT . . . FROM . . . UNION SELECT . . . FROM . . .

Recupera en una única consulta la información de varias. Los SELECTS deben ser compatibles.
Con la opción ALL se muestran las filas duplicadas.

F.- CLAÚSULA SELECT

SELECT [DISTINCT] { campos ¦ * } FROM tablas [WHERE condición] ;

Recupera registros de la base de datos con ciertos criterios.

G.- CLAÚSULA WHERE

SELECT campos FROM tablas WHERE criterio ;

Determina qué registros de las tablas deben enumerarse.

H.- CLAÚSULA GROUP BY – HAVING

1.- GROUP BY

SELECT campos [, funciones_columna] FROM tablas [WHERE condición]
GROUP BY campos ;

Los campos del GROUP BY deben aparecer en el SELECT.

Crea varias subtablas compuestas por filas con el mismo valor para la columna de agrupamiento.

2.- HAVING

SELECT campos [, funciones_columna] FROM tablas [WHERE condición]
GROUP BY campos HAVING condición

Los campos del GROUP BY deben aparecer en el SELECT.

Permite elegir grupos que se quieren visualizar.

I.- CONCATENADORES Y CUANTIFICADORES: ANY, ALL e IN

SELECT campos FROM tablas WHERE campo_concatenador {ANY ¦ ALL} (subselect) ;

ANY devuelve cierto si lo es para algún valor.

ALL devuelve cierto si lo es para todos los valores.

IN se usa si se sospecha que devuelve más de un valor (ver en apartado C de este anexo).

Sirven para anidar SELECT con SUBSELECTS. Son operaciones de relación y cláusulas IN.

Para operaciones de relación, el resultado el SUBSELECT debe ser único.

D.- OPERADORES LÓGICOS Y RELACIONALES:

1.- AND, OR, NOT: devuelven sus correspondientes valores de verdad lógicos.

2.- <, >, <>, >=, <=, = : idem

3.- BETWEEN: recupera registros según un intervalo de valores.
campo [NOT] BETWEEN valor_1 AND valor_2

4.- LIKE: compara una expresión de cadena con un modelo SQL
expresión LIKE ' modelo '
El modelo puede incluir: *, [carácter especial], ?, # (número), [-] (rango), ! (distinto).

5.- IN: recupera registros cuyo campo indicado coincide con alguno de los de la lista.
expresión [NOT] IN (valor_1, . . . , valor_n)

E.- CONSULTAS DE PREDICADO

* El predicado está entre la cláusula y los campos a recuperar.

1.- ALL (por defecto) : devuelve todos los campos de una tabla (equivale a *), pero es mejor seleccionar expresamente los campos para no ralentizar las consultas.

2.- TOP: devuelve un numero determinado de registros de una consulta order by.
[SELECT] TOP nº [PERCENT] campos FROM tabla
ORDER BY campos [ASC ¦ DESC];

PERCENT devuelve un porcentaje de los registros seleccionados.
ASC y DESC corresponden a los modos de ordenamiento ascendente y descendente.

3.- DISTINCT: omite registros con campos seleccionados coincidentes.

4.- DISTINCTROW: omite registros duplicados (todos los campos coincidentes).

5.- ALIAS: asigna otro nombre a un campo (equivale a un AS).
E.g: SELECT DISTINCTROW apellido AS empleado FROM plantilla

B.- RESTRICCIONES Y TRIGGERS

1.- RESTRICCIONES SOBRE UNA TABLA

CHECK condición ;

2.- RESTRICCIONES DE DOMINIO

CREATE DOMAIN nombre AS tipo ;

3.- ASERCIONES (RESTRICCIONES SOBRE VARIAS TABLAS)

CREATE ASSERTION nombre CHECK condicion ;

4.- TRIGGERS (o DISPARADORES)

CREATE TRIGGER nombre {BEFORE ¦ AFTER}
{DELETE ¦ INSERT ¦ UPDATE ¦ UPDATE OF campos }
[REFERENCING { OLD ¦ NEW} AS nombre]
{FOR EACH ROW ¦ FOR EACH STATEMENT }
ON tabla [WHEN (predicado)] BEGIN cuerpo END ;

C.- OPERADORES DE COLUMNA O AGRUPAMIENTO:

Se expresan detrás del SELECT o HAVING (nunca detrás del WHERE), y operan con todas las filas que cumplen la condición WHERE

{AVG ¦ COUNT ¦ SUM ¦ MAX ¦ MIN } campo

1.- AVG: promedio de los valores de un campo.

2.- COUNT: número de registros de una selección.
COUNT (*) : cuenta todos los registros, incluidos los nulos.

3.- SUM: suma de los valores de un campo.

4.- MAX, MIN: valor máximo o mínimo de los valores de un campo.

S.Q.L.

A.- OPERACIONES CON TABLAS

1.- CREAR TABLAS

CREATE TABLE tabla (campo_1 tipo_1 [NOT NULL], ... , campo_n tipo_n [NOT NULL]
[CONSTRAINT] alias PRIMARY KEY (campo_i), /* clave primaria */
[CONSTRAINT alias UNIQUE (campo_j), /* índice único */
[CONSTRAINT alias] CHECK (restricción), . . .] /* restricciones sobre una tabla */
CONSTRAINT alias FOREIGN KEY (clave_ajena)
REFERENCES tabla_referenciada (campo_ref)) ; /* clave ajena */

2.- BORRAR TABLAS

DROP TABLE tabla [SET NULL ¦ CASCADE ¦ RESTRICT];

3.- ACTUALIZAR TABLAS

AÑADIR UN CAMPO:
ALTER TABLE tabla ADD campo tipo ;

BORRAR UN CAMPO:
ALTER TABLE tabla DROP campo [SET NULL ¦ CASCADE ¦ RESTRICT];

4.- MANEJO DE DATOS (VALORES)

INSERTAR VALORES:
INSERT INTO tabla [(columna)] VALUES (expresiones) ;

MODIFICAR VALORES:
UPDATE tabla SET campo = expresión [WHERE condición];

BORRAR VALORES:
DELETE FROM tabla [WHERE condición];

* MODOS DE BORRADO Y ACTUALIZACIÓN:

SET NULL : pone a nulos los campos referenciados.
SET DEFAULT : pone al valor por defecto los campos referenciados.
CASCADE : borra incluso los campos referenciados.
RESTRICT : no borra si hay campos referenciados.